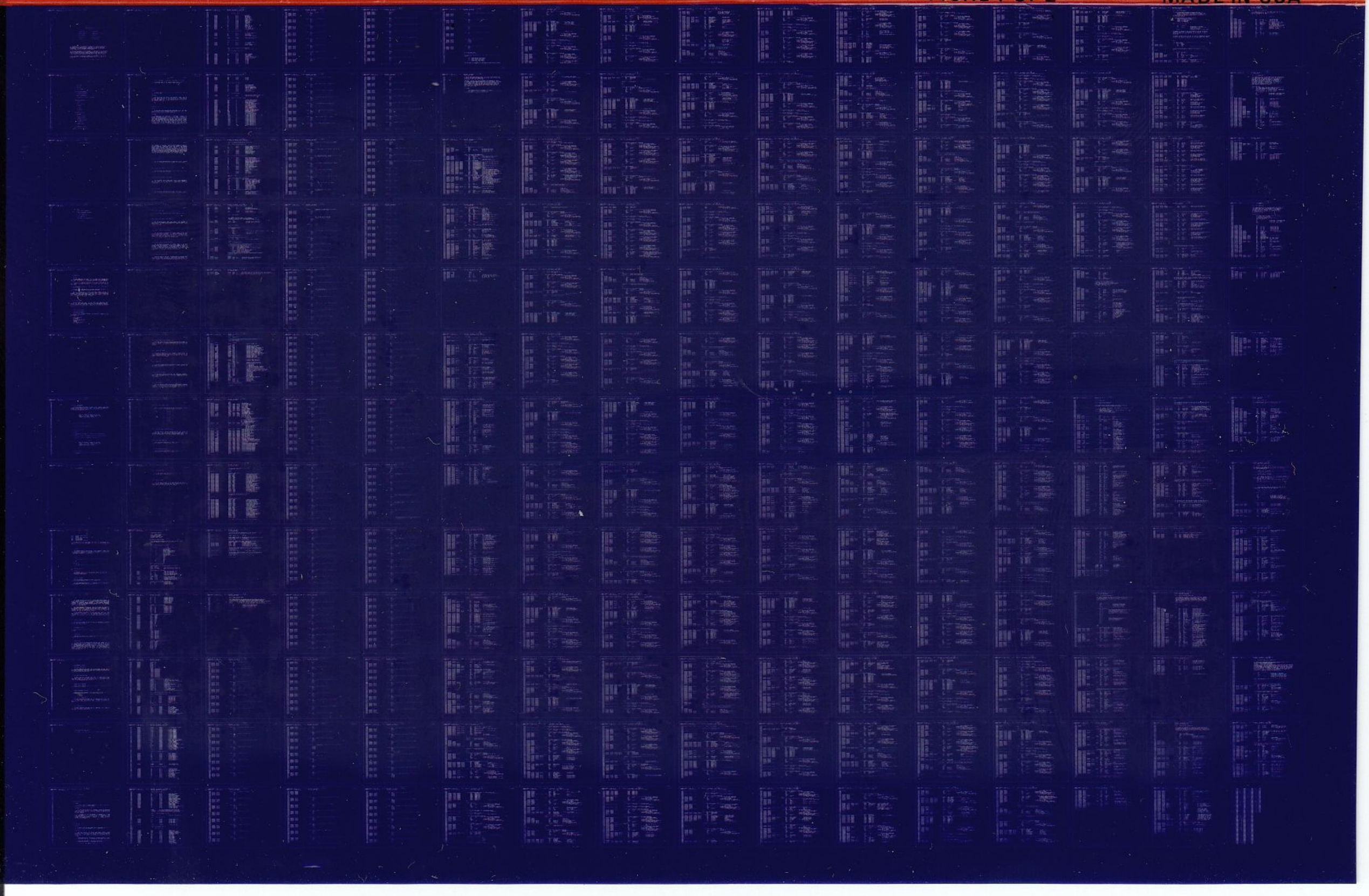
RM02,RM03

FCTNL TST 3
CZRMEC0

AH-B004C-MC

FICHE 1 OF 2

MAR 1979
digital
MADE IN USA



RM02,RM03

FCTNL TST 3

CZRMEC0

AH-B004C-MC

FICHE 2 OF 2

MAR 1979

digital

MADE IN USA

```
.REM    \
```

IDENTIFICATION

PRODUCT CODE:	AC-B003C-MC
PRODUCT NAME:	CZRMECO RM03/2 FCTNL TST 3
DATE CREATED:	FEB 1979
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

```
                .PAGE


                                        CONTENTS



```

108 2. DUAL PORT CONFIGURATIONS

3. MEMORY PARITY HARDWARE

4. MEMORY MANAGEMENT HARDWARE

5. ACT,APT COMPATIBILITY

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

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

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

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

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

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

1.2 UNIT UNDER TEST

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

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

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

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

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

2.2 MEDIA REQUIREMENTS

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

NOTE

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

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, CZRMJ-B

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

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

3.2 SWITCH OPTIONS

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

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

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

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

3.3 STARTING

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

3.4 HALTING

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

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

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

4.2 CONSOLE DIALOGUE

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

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

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

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

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

4.3 PROGRESS REPORTS

AN END OF PASS REPORT OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL DEVICES IN THE TEST GUE. 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

```
   466                                  NONEXISTENT DEVICE;
```

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

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

DRIVE TYPE TEST

PURPOSE:

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

PROCEDURE:

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

WRITE/READ 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

PAGE 34

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

```
                                                                F 2
   627                          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, PEAD W/CURRENT LEVEL SWITCHING

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

WRITE, READ EARLY PEAK SHIFT

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

WRITE, READ MIXED PEAK SHIFT

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

PAGE 38

WRITE, READ EACH TRACK

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

WRITE, READ W/ ABORT

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

```
                        ;PROGRAM REVISION #001

                        .TITLE  CZRMECO 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-DZGAC-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
```

```
811             000003                  R3=     %3              ;;GENERAL REGISTER
812             000004                  R4=     %4              ;;GENERAL REGISTER
813             000005                  R5=     %5              ;;GENERAL REGISTER
814             000006                  R6=     %6              ;;GENERAL REGISTER
815             000007                  R7=     %7              ;;GENERAL REGISTER
816             000006                  SP=     %6              ;;STACK POINTER
817             000007                  PC=     %7              ;;PROGRAM COUNTER
818
819                                     ;*PRIORITY LEVEL DEFINITIONS
820             000000                  PR0=    0               ;;PRIORITY LEVEL 0
821             000040                  PR1=    40              ;;PRIORITY LEVEL 1
822             000100                  PR2=    100             ;;PRIORITY LEVEL 2
823             000140                  PR3=    140             ;;PRIORITY LEVEL 3
824             000200                  PR4=    200             ;;PRIORITY LEVEL 4
825             000240                  PR5=    240             ;;PRIORITY LEVEL 5
826             000300                  PR6=    300             ;;PRIORITY LEVEL 6
827             000340                  PR7=    340             ;;PRIORITY LEVEL 7
828
829                                     ;*"SWITCH REGISTER" SWITCH DEFINITIONS
830             100000                  SW15=   100000
831             040000                  SW14=   40000
832             020000                  SW13=   20000
833             010000                  SW12=   10000
834             004000                  SW11=   4000
835             002000                  SW10=   2000
836             001000                  SW09=   1000
837             000400                  SW08=   400
838             000200                  SW07=   200
839             000100                  SW06=   100
840             000040                  SW05=   40
841             000020                  SW04=   20
842             000010                  SW03=   10
843             000004                  SW02=   4
844             000002                  SW01=   2
845             000001                  SW00=   1
846                                     .EQUIV  SW09,SW9
847                                     .EQUIV  SW08,SW8
848                                     .EQUIV  SW07,SW7
849                                     .EQUIV  SW06,SW6
850                                     .EQUIV  SW05,SW5
851                                     .EQUIV  SW04,SW4
852                                     .EQUIV  SW03,SW3
853                                     .EQUIV  SW02,SW2
854                                     .EQUIV  SW01,SW1
855                                     .EQUIV  SW00,SW0
856
857                                     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
858             100000                  BIT15=  100000
859             040000                  BIT14=  40000
860             020000                  BIT13=  20000
861             010000                  BIT12=  10000
862             004000                  BIT11=  4000
863             002000                  BIT10=  2000
864             001000                  BIT09=  1000
865             000400                  BIT08=  400
866             000200                  BIT07=  200
```

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

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

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

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

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

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

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

                        ;RMDA   DISK ADDRESS REGISTER

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

                        ;TRACK,SECTOR MASKS

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

                        ;RMDS   DRIVE STATUS REGISTER

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

```
                        ;RMER1  ERROR REGISTER #1
                        
100000                  DCK     =       BIT15           ;DATA CHECK ERROR
040000                  UNS     =       BIT14           ;DRIVE UNSAFE
020000                  OPI     =       BIT13           ;OPERATION INCOMPLETE
010000                  DTE     =       BIT12           ;DRIVE TIMING ERROR
004000                  WLE     =       BIT11           ;WRITE LOCK ERROR
002000                  IAE     =       BIT10           ;INVALID ADDRESS ERROR
001000                  AOE     =       BIT09           ;ADDRESS OVERFLOW ERROR
000400                  HCRC    =       BIT08           ;HEADER CRC ERROR
000200                  HCE     =       BIT07           ;HEADER COMPARE ERROR
000100                  ECH     =       BIT06           ;ECC "HARD" ERROR
000040                  WCF     =       BIT05           ;WRITE CLOCK FAILURE
000020                  FER     =       BIT04           ;FORMAT ERROR
000010                  PAR     =       BIT03           ;PARITY ERROR
000004                  RMR     =       BIT02           ;REGISTER MODIFICATION REFUSED
000002                  ILR     =       BIT01           ;ILLEGAL REGISTER
000001                  ILF     =       BIT00           ;ILLEGAL FUNCTION

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

                        ;RMAS   ATTENTION SUMMARY REGISTER

000377                  ATNMSK  =       377             ;MASK FOR ATTENTION BITS

                        ;RMLA   LOOK AHEAD REGISTER

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

003700                  SCTMSK  =       003700          ;SECTOR COUNT MASK

                        ;RMMR   MAINTENANCE REGISTER

                        ;       WRITE ONLY BITS

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

```
1035
1036                            ;       READ ONLY BITS
1037
1038    100000                  OCC     =       BIT15           ;OCCUPIED
1039    040000                  RG      =       BIT14           ;RUN AND GO
1040    020000                  EBL     =       BIT13           ;END OF BLOCK
1041    010000                  REX     =       BIT12           ;EXCEPTION
1042    004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1043    002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1044    001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1045    000400                  PDA     =       BIT08           ;DATA AREA
1046    000200                  PHA     =       BIT07           ;HEADER AREA
1047    000100                  CONT    =       BIT06           ;CONTINUE
1048    000040                  WC      =       BIT05           ;WORD CLOCK
1049    000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1050    000010                  MWD     =       BIT03           ;WRITE DATA BIT
1051    000004                  LS      =       BIT02           ;LAST SECTOR
1052    000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1053    000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1054
1055                            ;RMDT   DRIVE TYPE REGISTER
1056
1057    100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1058    040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1059    020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1060    004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1061
1062    020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1063    024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1064
1065                            ;RMOF   OFFSET REGISTER
1066
1067    010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1068    004000                  ECI     =       BIT11           ;ECC INHIBIT
1069    002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1070    000200                  OFD     =       BIT07           ;OFFSET FORWARD
1071
1072
1073                            ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1074    001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1075
1076                            ;RMMR2  MAINTENANCE REGISTER #2
1077
1078                            ;       READ ONLY BITS
1079    100000                  RQA     =       BIT15           ;PORT A REQUEST
1080    040000                  RQB     =       BIT14           ;PORT B REQUEST
1081    020000                  TAG     =       BIT13           ;TAG CONTROL
1082    010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1083    004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1084    002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1085    001000                  BB09    =       BIT09           ;TAG BUS
1086    000400                  BB08    =       BIT08           ;TAG BUS
1087    000200                  BB07    =       BIT07           ;TAG BUS
1088    000100                  BB06    =       BIT06           ;TAG BUS
1089    000040                  BB05    =       BIT05           ;TAG BUS
1090    000020                  BB04    =       BIT04           ;TAG BUS
```

```
1091            000010                  BB03    =       BIT03           ;TAG BUS
1092            000004                  BB02    =       BIT02           ;TAG BUS
1093            000002                  BB01    =       BIT01           ;TAG BUS
1094            000001                  BB00    =       BIT00           ;TAG BUS
1095
1096
1097                                    ;RMER2  ERROR REGISTER 2
1098
1099            100000                  BSE     =       BIT15           ;BAD SECTOR ERROR
1100            040000                  SKI     =       BIT14           ;SEEK INCOMPLETE
1101            020000                  OPE     =       BIT13           ;OPERATOR PLUG ERROR
1102            010000                  IVC     =       BIT12           ;INVALID COMMAND ERROR
1103            004000                  LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
1104            002000                  LBC     =       BIT10           ;LOSS OF BIT CLOCK
1105            000200                  DVC     =       BIT07           ;DEVICE CHECK
1106            000010                  DPE     =       BIT03           ;DATA PARITY ERROR
1107
1108                                    .SBTTL  PROGRAM MNEMONICS
1109
1110            100000                  MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
1111            040000                  USE     =       BIT14           ;USER DETECTED SECTOR ERROR
1112
1113                                    .SBTTL  RM03 REGISTER INDEX VALUES
1114
1115            000000                  RMCS1   =       00              ;CONTROL STATUS REGISTER
1116            000006                  RMDA    =       06              ;DISK ADDRESS REGISTER
1117            000012                  RMDS    =       12              ;DRIVE STATUS REGISTER
1118            000014                  RMER1   =       14              ;ERROR REGISTER 1
1119            000016                  RMAS    =       16              ;ATTENTION SUMMARY REGISTER
1120            000020                  RMLA    =       20              ;LOOK AHEAD REGISTER
1121            000024                  RMMR1   =       24              ;MAINTENANCE REGISTER
1122            000026                  RMDT    =       26              ;DRIVE TYPE REGISTER
1123            000030                  RMSN    =       30              ;SERIAL NUMBER REGISTER
1124            000032                  RMOF    =       32              ;OFFSET REGISTER
1125            000034                  RMDC    =       34              ;DESIRED CYLINDER REGISTER
1126            000036                  RMCC    =       36              ;CURRENT CYLINDER REGISTER
1127            000040                  RMMR2   =       40              ;MAINTENANCE REGISTER 2
1128            000042                  RMER2   =       42              ;ERROR REGISTER 2
1129            000044                  RMEC1   =       44              ;ECC POSITION REGISTER
1130            000046                  RMEC2   =       46              ;ECC PATTERN REGISTER
1131            000050                  ILRG50  =       50              ;ILLEGAL REGISTER 50
1132            000052                  ILRG52  =       52              ;ILLEGAL REGISTER 52
1133            000054                  ILRG54  =       54              ;ILLEGAL REGISTER 54
1134            000056                  ILRG56  =       56              ;ILLEGAL REGISTER 56
1135            000060                  ILRG60  =       60              ;ILLEGAL REGISTER 60
1136            000062                  ILRG62  =       62              ;ILLEGAL REGISTER 62
1137            000064                  ILRG64  =       64              ;ILLEGAL REGISTER 64
1138            000066                  ILRG66  =       66              ;ILLEGAL REGISTER 66
1139            000070                  ILRG70  =       70              ;ILLEGAL REGISTER 70
1140            000072                  ILRG72  =       72              ;ILLEGAL REGISTER 72
1141            000074                  ILRG74  =       74              ;ILLEGAL REGISTER 74
1142            000076                  ILRG76  =       76              ;ILLEGAL REGISTER 76
1143
1144
1145            000077                  IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
1146
```

```
1147                            .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
1148
1149                            ;RMCS1  CONTROL STATUS REGISTER #1
1150
1151    100000                  SC      =       BIT15           ;SPECIAL CONDITION-READ ONLY
1152    040000                  TRE     =       BIT14           ;TRANSFER ERROR
1153    020000                  MCPE    =       BIT13           ;MASSBUS CONTROL BUS PARITY
1154                                                            ;ERROR-READ ONLY
1155    002000                  PSEL    =       BIT10           ;PORT B SELECT
1156    001000                  A17     =       BIT09           ;ADDRESS EXTENSION
1157    000400                  A16     =       BIT08           ;ADDRESS EXTENSION
1158    000200                  RDY     =       BIT07           ;READY-READ ONLY
1159    000100                  IE      =       BIT06           ;INTERRUPT ENABLE
1160
1161                            ;RMCS2  RH CONTROL STATUS REGISTER #2
1162
1163    100000                  DLT     =       BIT15           ;DATA LATE-READ ONLY
1164    040000                  WCE     =       BIT14           ;WRITE CHECK ERROR-READ ONLY
1165    020000                  UPE     =       BIT13           ;UNIBUS PARITY ERROR
1166    010000                  NED     =       BIT12           ;NONEXISTANT DRIVE-READ ONLY
1167    004000                  NEM     =       BIT11           ;NONEXISTANT MEMORY-READ ONLY
1168    002000                  PGE     =       BIT10           ;PROGRAM ERROR-READ ONLY
1169    001000                  MXF     =       BIT09           ;MISSED TRANSFER
1170    000400                  MDPE    =       BIT08           ;MASSBUS DATA BUS PARITY
1171                                                            ;ERROR-READ ONLY
1172    000200                  OR      =       BIT07           ;OUTPUT READY-READ ONLY
1173    000100                  IR      =       BIT06           ;INPUT READY-READ ONLY
1174    000040                  CLR     =       BIT05           ;CONTROLLER CLEAR
1175    000020                  PAT     =       BIT04           ;PARITY TEST
1176    000010                  BAI     =       BIT03           ;UNIBUS ADDRESS INCREMENT
1177                                                            ;INHIBIT
1178    000004                  U2      =       BIT02           ;UNIT SELECT
1179    000002                  U1      =       BIT01           ;UNIT SELECT
1180    000001                  U0      =       BIT00           ;UNIT SELECT
1181
1182                            ;UNIT SELECT MASK
1183    000007                  UNTMSK  =       7               ;UNIT SELECT MASK
1184
1185                            ;RMCS3  RH70 CONTROL STATUS REGISTER #3
1186    100000                  APE     =       BIT15           ;ADDRESS PARITY ERROR
1187    040000                  DPEHI   =       BIT14           ;DATA PARITY ERROR HIGH WORD
1188    020000                  DPELO   =       BIT13           ;DATA PARITY ERROR LOW WORD
1189    010000                  WCEHI   =       BIT12           ;WRITE CHECK ERROR HIGH WORD
1190    004000                  WCELO   =       BIT11           ;WRITE CHECK ERROR LOW WORD
1191    002000                  DBL     =       BIT10           ;DOUBLE WORD TRANSFER
1192    000100                  IE      =       BIT06           ;INTERRUPT ENABLE
1193    000010                  IPCK3   =       BIT03           ;INVERT PARITY CHECK
1194    000004                  IPCK2   =       BIT02           ;INVERT PARITY CHECK
1195    000002                  IPCK1   =       BIT01           ;INVERT PARITY CHECK
1196    000001                  IPCK0   =       BIT00           ;INVERT PARITY CHECK
1197                            .SBTTL  RH CONTROLLER REGISTER INDEX VALUES
1198
1199    000000                  RMCS1   =       00              ;CONTROL, STATUS REGISTER
1200    000002                  RMWC    =       02              ;WORD COUNT REGISTER
1201    000004                  RMBA    =       04              ;BUS ADDRESS REGISTER
1202    000010                  RMCS2   =       10              ;CONTROLLER STATUS REGISTER
```

```
1203            000022                  RMDB    =       22              ;DATA BUFFER
1204            000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
1205            000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3
1206
1207            176700                  ABASE   =       176700          ;UNIBUS ADDRESS
1208            120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY
1209
1210
1211                                    .SBTTL  TRAP CATCHER
1212
1213            000000                          .=0
1214                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1215                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1216                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1217            000174                  .=174
1218  000174    000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
1219  000176    000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
1220
1221
1222                                    .SBTTL  ACT11 HOOKS
1223
1224                                    ;;******************************************************************
1225                                    ;HOOKS REQUIRED BY ACT11
1226            000200                          $SVPC=.                 ;SAVE PC
1227            000046                          .=46
1228  000046    037176                        $ENDAD                    ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1229            000052                          .=52
1230  000052    000000                        .WORD   0                 ;;2)SET LOC.52 TO ZERO
1231            000200                        .=$SVPC                   ;; RESTORE PC
1232
1233                                    .SBTTL  STARTING ADDRESS
1234
1235                                    ;THE PROGRAM STARTS AT LOCATION 200
1236            000200                  .       =       200
1237  000200    000137  005324                JMP     START           ;JUMP TO START OF PROGRAM
1238
1239            001100                  .=1100
1240                                    .SBTTL  APT PARAMETER BLOCK
1241
1242                                    ;;******************************************************************
1243                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1244                                    ;;******************************************************************
1245            001100                          .$X=.   ;;SAVE CURRENT LOCATION
1246            000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1247  000024    000200                        200     ;;FOR APT START UP
1248            000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1249  000044    001100                        $APTHDR ;;POINT TO APT HEADER BLOCK
1250            001100                          .=.$X   ;;RESET LOCATION COUNTER
1251                                    ;;******************************************************************
1252                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1253                                    ;INTERFACE SPEC.
1254
1255  001100                            $APTHD:
1256  001100    000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1257  001102    001222                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1258  001104    000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

```
1259  001106  000002                $PASTM: .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1260  001110  000002                $UNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1261  001112  000042                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1262          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

                                ;;**********************************************************************
                                .NLIST ME
                                ;
                                ;
```

```
1319                                    .EVEN
1320  001222                            $MAIL:                          ;;APT MAILBOX
1321  001222  000000                    $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
1322  001224  000000                    $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
1323  001226  000000                    $TESTN: .WORD   ATESTN  ;;TEST NUMBER
1324  001230  000000                    $PASS:  .WORD   APASS   ;;PASS COUNT
1325  001232  000000                    $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1326  001234  000000                    $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1327  001236  000000                    $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1328  001240  000000                    $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1329  001242                            $ETABLE:                        ;;APT ENVIRONMENT TABLE
1330  001242     000                    $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1331  001243     000                    $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1332  001244  000000                    $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1333  001246  000000                    $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1334  001250  000000                    $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1335                                    ;*                      BITS 15-11=CPU TYPE
1336                                    ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1337                                    ;*                              11/70=06,PDQ=07,Q=10
1338                                    ;*                      BIT 10=REAL TIME CLOCK
1339                                    ;*                      BIT  9=FLOATING POINT PROCESSOR
1340                                    ;*                      BIT  8=MEMORY MANAGEMENT
1341  001252     000                    $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1342  001253     000                    $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1343                                    ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1344                                    ;*                              900 NSEC CORE=001
1345                                    ;*                              300 NSEC BIPOLAR=002
1346                                    ;*                              500 NSEC MOS=003
1347  001254  000000                    $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1348                                    ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
1349  001256     000                    $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1350  001257     000                    $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1351  001260  000000                    $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1352  001262     000                    $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1353  001263     000                    $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1354  001264  000000                    $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1355  001266     000                    $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1356  001267     000                    $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1357  001270  000000                    $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1358  001272  120254                    $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1359  001274  000000                    $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1360  001276  176700                    $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1361  001300  000000                    $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1362  001302  000000                    $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1363  001304  000000                    $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1364  001306  000000                    $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
1365  001310  000000                    $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
1366  001312  000000                    $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
1367  001314  000000                    $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
1368  001316  000000                    $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
1369  001320  000000                    $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
1370  001322  000000                    $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1371  001324  000000                    $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1372  001326                            $ETEND:
1373                                    .MEXIT
1374  001326  000000                    CTLFG:  .WORD   0               ;CONTROL-C FLAG
```

```
1375
1376                                    ;THE REGISTER INPUT BUFFER IS USED FOR
1377                                    ;STORING DRIVE STATUS
1378
1379  001330                            GETBUF:
1380
1381                                    ;REGISTER INPUT BUFFER
1382  001330  000000                    RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER
1383  001332  000000                    RMWCI: .WORD                    ;WORD COUNT REGISTER
1384  001334  000000                    RMBAI: .WORD                    ;BUS ADDRESS REGISTER
1385  001336  000000                    RMDAI: .WORD                    ;DISK ADDRESS REGISTER
1386  001340  000000                    RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
1387  001342  000000                    RMDSI: .WORD                    ;DRIVE STATUS REGISTER
1388  001344  000000                    RMER1I: .WORD                   ;ERROR REGISTER 1
1389  001346  000000                    RMASI: .WORD                    ;ATTENTION SUMMARY REGISTER
1390  001350  000000                    RMLAI: .WORD                    ;LOOK AHEAD REGISTER
1391  001352  000000                    RMDBI: .WORD                    ;DATA BUFFER
1392  001354  000000                    RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
1393  001356  000000                    RMDTI: .WORD                    ;DRIVE TYPE REGISTER
1394  001360  000000                    RMSNI: .WORD                    ;SERIAL NUMBER REGISTER
1395  001362  000000                    RMOFI: .WORD                    ;OFFSET REGISTER
1396  001364  000000                    RMDCI: .WORD                    ;DESIRED CYLINDER REGISTER
1397  001366  000000                    RMCCI: .WORD                    ;CURRENT CYLINDER REGISTER
1398  001370  000000                    RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
1399  001372  000000                    RMER2I: .WORD                   ;ERROR REGISTER 2
1400  001374  000000                    RMEC1I: .WORD                   ;ECC POSITION REGISTER
1401  001376  000000                    RMEC2I: .WORD                   ;ECC PATTERN REGISTER
1402
1403                                    ;THE REGISTEP OUTPUT BUFFER IS USED FOR
1404                                    ;ASSEMBLING DATA GOING TO REGISTER
1405
1406  001400                            PUTBUF:
1407
1408                                    ;REGISTER OUTPUT BUFFER
1409  001400  000000                    RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER
1410  001402  000000                    RMWCO: .WORD                    ;WORD COUNT REGISTER
1411  001404  000000                    RMBAO: .WORD                    ;BUS ADDRESS REGISTER
1412  001406  000000                    RMDAO: .WORD                    ;DISK ADDRESS REGISTER
1413  001410  000000                    RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
1414  001412  000000                    RMDSO: .WORD                    ;DRIVE STATUS REGISTER
1415  001414  000000                    RMER1O: .WORD                   ;ERROR REGISTER 1
1416  001416  000000                    RMASO: .WORD                    ;ATTENTION SUMMARY REGISTER
1417  001420  000000                    RMLAO: .WORD                    ;LOOK AHEAD REGISTER
1418  001422  000000                    RMDBO: .WORD                    ;DATA BUFFER
1419  001424  000000                    RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
1420  001426  000000                    RMDTO: .WORD                    ;DRIVE TYPE REGISTER
1421  001430  000000                    RMSNO: .WORD                    ;SERIAL NUMBER REGISTER
1422  001432  000000                    RMOFO: .WORD                    ;OFFSET REGISTER
1423  001434  000000                    RMDCO: .WORD                    ;DESIRED CYLINDER REGISTER
1424  001436  000000                    RMCCO: .WORD                    ;CURRENT CYLINDER REGISTER
1425  001440  000000                    RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
1426  001442  000000                    RMER2O: .WORD                   ;ERROR REGISTER 2
1427  001444  000000                    RMEC1O: .WORD                   ;ECC POSITION REGISTER
1428  001446  000000                    RMEC2O: .WORD                   ;ECC PATTERN REGISTER
1429
1430                                    ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
```

```
1431                                    ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
1432                                    ;WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.
1433
1434  001450  000012                    TSTQUE: .BLKW   10.     ;TEST QUE
1435
1436                                    ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
1437                                    ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
1438  001474  000000                    MEDENB: .WORD           ;MEDIA ENABLE
1439
1440                                    ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
1441                                    ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
1442  001476  000000                    ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
1443  001500  000000                    ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
1444  001502  000000                    CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
1445  001504  000000                    CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
1446
1447                                    ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
1448                                    ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
1449                                    ;A NEGATIVE BYTE.
1450  001506  000027                    GETINX: .B KB   23.             ;GET INDEX TABLE
1451
1452                                    ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
1453                                    ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
1454                                    ;A NEGATIVE BYTE.
1455  001535  000027                    PUTINX: .BLKB   23.             ;PUT INDEX TABLE
1456
1457                                    ;PUT TAGS HERE
1458
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001564                          $ERRTB:
```

```
1476
1477  001564  071432                    ;ERROR  1       WRONG UNIT SELECTED
1478  001566  075524                            EMT1
1479  001570  075650                            EHT1
1480  001572  075740                            EDT1
1481                                      EFT1
1482
1483                                      ;ERROR  2       DEVICE WENT UNAVAILABLE
1484  001574  071436                            EMT2
1485  001576  075524                            EHT1
1486  001600  075650                            EDT1
1487  001602  075740                            EFT1
1488
1489
1490                                      ;ERROR  3       DEVICE WENT NONEXISTENT
1491  001604  071444                            EMT3
1492  001606  075524                            EHT1
1493  001610  075650                            EDT1
1494  001612  075740                            EFT1
1495
1496
1497                                      ;ERROR  4       CONTROLLER NOT READY
1498  001614  071452                            EMT4
1499  001616  075524                            EHT1
1500  001620  075650                            EDT1
1501  001622  075740                            EFT1
1502
1503
1504                                      ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1505  001624  071460                            EMT5
1506  001626  075524                            EHT1
1507  001630  075650                            EDT1
1508  001632  075740                            EFT1
1509
1510
1511                                      ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1512  001634  071466                            EMT6
1513  001636  075524                            EHT1
1514  001640  075650                            EDT1
1515  001642  075740                            EFT1
1516
1517
1518                                      ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1519  001644  071474                            EMT7
1520  001646  000000                            0
1521  001650  000000                            0
1522  001652  000000                            0
1523
1524
1525                                      ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
1526  001654  071502                            EMT10
1527  001656  075524                            EHT1
1528  001660  075650                            EDT1
1529  001662  075740                            EFT1
1530
1531
```

```
1532
1533  001664  071506                    ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
1534  001666  075524                            EMT11
1535  001670  075650                            EHT1
1536  001672  075740                            EDT1
1537                                      EFT1
1538
1539                                      ;ERROR  12      INCORRECT FUNCTION CODE
1540  001674  071512                            EMT12
1541  001676  075524                            EHT1
1542  001700  075650                            EDT1
1543  001702  075740                            EFT1
1544
1545
1546                                      ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1547  001704  071520                            EMT13
1548  001706  075524                            EHT1
1549  001710  075650                            EDT1
1550  001712  075740                            EFT1
1551
1552
1553                                      ;ERROR  14      TRANSFER ERROR IS INCORRECT
1554  001714  071532                            EMT14
1555  001716  075524                            EHT1
1556  001720  075650                            EDT1
1557  001722  075740                            EFT1
1558
1559
1560                                      ;ERROR  15      INCORRECT WORD COUNT
1561  001724  071540                            EMT15
1562  001726  075524                            EHT1
1563  001730  075650                            EDT1
1564  001732  075740                            EFT1
1565
1566
1567                                      ;ERROR  16      INCORRECT BUS ADDRESS
1568  001734  071546                            EMT16
1569  001736  075524                            EHT1
1570  001740  075650                            EDT1
1571  001742  075740                            EFT1
1572
1573
1574                                      ;ERROR  17      INCORRECT LBT STATUS
1575  001744  071556                            EMT17
1576  001746  075524                            EHT1
1577  001750  075650                            EDT1
1578  001752  075740                            EFT1
1579
1580
1581                                      ;ERROR  20      INCORRECT AOE
1582  001754  071566                            EMT20
1583  001756  075524                            EHT1
1584  001760  075650                            EDT1
1585  001762  075740                            EFT1
1586
1587
```

```
1588
1589  001764  071576                    ;ERROR  21      INCORRECT DISK ADDRESS
1590  001766  075524                            EMT21
1591  001770  075650                            EHT1
1592  001772  075740                            EDT1
1593                                      EFT1
1594
1595                                      ;ERROR  22      INCORRECT CYLINDER ADDRESS
1596  001774  071606                            EMT22
1597  001776  075524                            EHT1
1598  002000  075650                            EDT1
1599  002002  075740                            EFT1
1600
1601
1602                                      ;ERROR  23      INCORRECT WLE STATUS
1603  002004  071616                            EMT23
1604  002006  075524                            EHT1
1605  002010  075650                            EDT1
1606  002012  075740                            EFT1
1607
1608
1609                                      ;ERROR  24      INCORRECT UPE STATUS
1610  002014  071626                            EMT24
1611  002016  075524                            EHT1
1612  002020  075650                            EDT1
1613  002022  075740                            EFT1
1614
1615
1616                                      ;ERROR  25      INCORRECT WCF STATUS
1617  002024  071636                            EMT25
1618  002026  075524                            EHT1
1619  002030  075650                            EDT1
1620  002032  075740                            EFT1
1621
1622
1623                                      ;ERROR  26      INCORRECT WCE STATUS
1624  002034  071646                            EMT26
1625  002036  075524                            EHT1
1626  002040  075650                            EDT1
1627  002042  075740                            EFT1
1628
1629
1630                                      ;ERROR  27      INCORRECT MDPE STATUS
1631  002044  071656                            EMT27
1632  002046  075524                            EHT1
1633  002050  075650                            EDT1
1634  002052  075740                            EFT1
1635
1636
1637                                      ;ERROR  30      INCORRECT DCK STATUS
1638  002054  071666                            EMT30
1639  002056  075524                            EHT1
1640  002060  075650                            EDT1
1641  002062  075740                            EFT1
1642
1643
```

```
1644
1645  002064  071676                    ;ERROR  31      INCORRECT ECH STATUS
1646  002066  075524                            EMT31
1647  002070  075650                            EHT1
1648  002072  075740                            EDT1
1649                                      EFT1
1650
1651                                      ;ERROR  32      DLT SHOULD NOT BE SET
1652  002074  071706                            EMT32
1653  002076  075524                            EHT1
1654  002100  075650                            EDT1
1655  002102  075740                            EFT1
1656
1657
1658                                      ;ERROR  33      MXF SHOULD NOT BE SET
1659  002104  071716                            EMT33
1660  002106  075524                            EHT1
1661  002110  075650                            EDT1
1662  002112  075740                            EFT1
1663
1664
1665                                      ;ERROR  34      DTE SHOULD NOT BE SET
1666  002114  071726                            EMT34
1667  002116  075524                            EHT1
1668  002120  075650                            EDT1
1669  002122  075740                            EFT1
1670
1671
1672                                      ;ERROR  35      INCORRECT HCRC STATUS
1673  002124  071736                            EMT35
1674  002126  075524                            EHT1
1675  002130  075650                            EDT1
1676  002132  075740                            EFT1
1677
1678
1679                                      ;ERROR  36      INCORRECT HCE STATUS
1680  002134  071746                            EMT36
1681  002136  075524                            EHT1
1682  002140  075650                            EDT1
1683  002142  075740                            EFT1
1684
1685
1686                                      ;ERROR  37      INCORRECT FER STATUS
1687  002144  071756                            EMT37
1688  002146  075524                            EHT1
1689  002150  075650                            EDT1
1690  002152  075740                            EFT1
1691
1692
1693                                      ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1694  002154  071766                            EMT40
1695  002156  075524                            EHT1
1696  002160  075650                            EDT1
1697  002162  075740                            EFT1
1698
1699
```

```
1700
1701  002164  071774                    ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1702  002166  075524                            EMT41
1703  002170  075650                            EHT1
1704  002172  075740                            EDT1
1705                                      EFT1
1706
1707                                      ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1708  002174  072004                            EMT42
1709  002176  075524                            EHT1
1710  002200  075650                            EDT1
1711  002202  075740                            EFT1
1712
1713
1714                                      ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1715  002204  072016                            EMT43
1716  002206  075524                            EHT1
1717  002210  075650                            EDT1
1718  002212  075740                            EFT1
1719
1720
1721                                      ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1722  002214  072026                            EMT44
1723  002216  075524                            EHT1
1724  002220  075650                            EDT1
1725  002222  075740                            EFT1
1726
1727
1728                                      ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1729  002224  072036                            EMT45
1730  002226  075524                            EHT1
1731  002230  075650                            EDT1
1732  002232  075740                            EFT1
1733
1734
1735                                      ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
1736  002234  072046                            EMT46
1737  002236  075524                            EHT1
1738  002240  075650                            EDT1
1739  002242  075740                            EFT1
1740
1741
1742                                      ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1743  002244  072056                            EMT47
1744  002246  075524                            EHT1
1745  002250  075650                            EDT1
1746  002252  075740                            EFT1
1747
1748
1749                                      ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1750  002254  072064                            EMT50
1751  002256  075524                            EHT1
1752  002260  075650                            EDT1
1753  002262  075740                            EFT1
1754
1755
```

```
1756
1757  002264  072074                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1758  002266  075524                            EMT51
1759  002270  075650                            EHT1
1760  002272  075740                            EDT1
1761                                      EFT1
1762
1763                                      ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1764  002274  072106                            EMT52
1765  002276  075524                            EHT1
1766  002300  075650                            EDT1
1767  002302  075740                            EFT1
1768
1769
1770                                      ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1771                                      ;               ON CYLINDER LATCH DIDN'T RESET
1772  002304  072124                            EMT53
1773  002306  075524                            EHT1
1774  002310  075650                            EDT1
1775  002312  075740                            EFT1
1776
1777
1778                                      ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1779  002314  072142                            EMT54
1780  002316  075524                            EHT1
1781  002320  075650                            EDT1
1782  002322  075740                            EFT1
1783
1784
1785                                      ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1786  002324  072152                            EMT55
1787  002326  075524                            EHT1
1788  002330  075650                            EDT1
1789  002332  075740                            EFT1
1790
1791
1792                                      ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1793  002334  072164                            EMT56
1794  002336  075524                            EHT1
1795  002340  075650                            EDT1
1796  002342  075740                            EFT1
1797
1798
1799                                      ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1800  002344  072202                            EMT57
1801  002346  075524                            EHT1
1802  002350  075650                            EDT1
1803  002352  075740                            EFT1
1804
1805
1806                                      ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1807                                      ;               VOLUME VALID
1808  002354  072212                            EMT60
1809  002356  075524                            EHT1
1810  002360  075650                            EDT1
1811  002362  075740                            EFT1
```

```
1812
1813
1814                                    ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
1815                                    ;               VOLUME VALID IS STIL SET
1816  002364  072230                            EMT61
1817  002366  075524                            EHT1
1818  002370  075650                            EDT1
1819  002372  075740                            EFT1
1820
1821
1822                                    ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
1823                                    ;               REPORTED DURING SEEK COMMAND
1824  002374  072250                            EMT62
1825  002376  075524                            EHT1
1826  002400  075650                            EDT1
1827  002402  075740                            EFT1
1828
1829
1830                                    ;ERROR  63      UNUSED
1831  002404  000000                            0
1832  002406  000000                            0
1833  002410  000000                            0
1834  002412  000000                            0
1835
1836
1837                                    ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
1838  002414  072266                            EMT64
1839  002416  075524                            EHT1
1840  002420  075650                            EDT1
1841  002422  075740                            EFT1
1842
1843
1844                                    ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
1845  002424  072306                            EMT65
1846  002426  075524                            EHT1
1847  002430  075650                            EDT1
1848  002432  075740                            EFT1
1849
1850
1851                                    ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
1852  002434  072326                            EMT66
1853  002436  075524                            EHT1
1854  002440  075650                            EDT1
1855  002442  075740                            EFT1
1856
1857
1858                                    ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
1859  002444  072340                            EMT67
1860  002446  075524                            EHT1
1861  002450  075650                            EDT1
1862  002452  075740                            EFT1
1863
1864
1865                                    ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
1866  002454  072352                            EMT70
1867  002456  075524                            EHT1
```

```
1868  002460  075650                    EDT1
1869  002462  075740                    EFT1
1870
1871
1872                                    ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
1873  002464  072362                    EMT71
1874  002466  075524                    EHT1
1875  002470  075650                    EDT1
1876  002472  075740                    EFT1
1877
1878
1879                                    ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
1880  002474  072372                    EMT72
1881  002476  075524                    EHT1
1882  002500  075650                    EDT1
1883  002502  075740                    EFT1
1884
1885
1886                                    ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
1887                                    ;               CYLINDER DIDNT DROP
1888  002504  072410                    EMT73
1889  002506  075524                    EHT1
1890  002510  075650                    EDT1
1891  002512  075740                    EFT1
1892
1893
1894                                    ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
1895  002514  072426                    EMT74
1896  002516  075524                    EHT1
1897  002520  075650                    EDT1
1898  002522  075740                    EFT1
1899
1900
1901                                    ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
1902  002524  072436                    EMT75
1903  002526  075524                    EHT1
1904  002530  075650                    EDT1
1905  002532  075740                    EFT1
1906
1907
1908                                    ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
1909  002534  072456                    EMT76
1910  002536  075524                    EHT1
1911  002540  075650                    EDT1
1912  002542  075740                    EFT1
1913
1914
1915                                    ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
1916  002544  072466                    EMT77
1917  002546  075524                    EHT1
1918  002550  075650                    EDT1
1919  002552  075740                    EFT1
1920
1921
1922                                    ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
1923  002554  072500                    EMT100
```

```
1924  002556  075524                        EMT1
1925  002560  075650                        EDT1
1926  002562  075740                        EFT1
1927
1928
1929                              ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
1930  002564  072516                        EMT101
1931  002566  075524                        EMT1
1932  002570  075650                        EDT1
1933  002572  075740                        EFT1
1934
1935
1936                              ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
1937  002574  072534                        EMT102
1938  002576  075524                        EMT1
1939  002600  075650                        EDT1
1940  002602  075740                        EFT1
1941
1942
1943                              ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
1944  002604  072544                        EMT103
1945  002606  075524                        EMT1
1946  002610  075650                        EDT1
1947  002612  075740                        EFT1
1948
1949
1950                              ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
1951  002614  072556                        EMT104
1952  002616  075524                        EMT1
1953  002620  075650                        EDT1
1954  002622  075740                        EFT1
1955
1956
1957                              ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
1958  002624  072574                        EMT105
1959  002626  075524                        EMT1
1960  002630  075650                        EDT1
1961  002632  075740                        EFT1
1962
1963
1964                              ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
1965  002634  072604                        EMT106
1966  002636  075524                        EMT1
1967  002640  075650                        EDT1
1968  002642  075740                        EFT1
1969
1970
1971                              ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
1972  002644  072614                        EMT107
1973  002646  075524                        EMT1
1974  002650  075650                        EDT1
1975  002652  075740                        EFT1
1976
1977
1978                              ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
1979  002654  072634                        EMT110
```

```
1980  002656  075546                        EMT110
1981  002660  075666                        EDT110
1982  002662  075756                        EFT110
1983
1984
1985                                ;ERROR  111     NONEXISTENT DEVICE
1986  002664  072646                        EMT111
1987  002666  075552                        EHT111
1988  002670  075670                        EDT111
1989  002672  075760                        EFT111
1990
1991
1992                                ;ERROR  112     DEVICE NOT AVAILABLE
1993  002674  072654                        EMT112
1994  002676  075552                        EHT111
1995  002700  075670                        EDT111
1996  002702  075760                        EFT111
1997
1998
1999                                ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
2000  002704  072662                        EMT113
2001  002706  000000                        0
2002  002710  000000                        0
2003  002712  000000                        0
2004
2005
2006                                ;ERROR  114     DEVICE NOT AN RM03
2007  002714  072676                        EMT114
2008  002716  075556                        EHT114
2009  002720  075672                        EDT114
2010  002722  075762                        EFT114
2011
2012
2013                                ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
2014  002724  072704                        EMT115
2015  002726  075524                        EHT1
2016  002730  075650                        EDT1
2017  002732  075740                        EFT1
2018
2019
2020                                ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
2021  002734  072714                        EMT116
2022  002736  075524                        EHT1
2023  002740  075650                        EDT1
2024  002742  075740                        EFT1
2025
2026
2027                                ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
2028  002744  072724                        EMT117
2029  002746  075524                        EHT1
2030  002750  075650                        EDT1
2031  002752  075740                        EFT1
2032
2033
2034                                ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
2035  002754  072734                        EMT120
```

```
2036  002756  075524                    EHT1
2037  002760  075650                    EDT1
2038  002762  075740                    EFT1
2039
2040
2041                              ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
2042  002764  072744                    EMT121
2043  002766  075524                    EHT1
2044  002770  075650                    EDT1
2045  002772  075740                    EFT1
2046
2047
2048                              ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
2049  002774  072754                    EMT122
2050  002776  075524                    EHT1
2051  003000  075650                    EDT1
2052  003002  075740                    EFT1
2053
2054
2055                              ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
2056  003004  072764                    EMT123
2057  003006  075524                    EHT1
2058  003010  075650                    EDT1
2059  003012  075740                    EFT1
2060
2061
2062                              ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
2063  003014  072774                    EMT124
2064  003016  075524                    EHT1
2065  003020  075650                    EDT1
2066  003022  075740                    EFT1
2067
2068
2069                              ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
2070  003024  073004                    EMT125
2071  003026  075524                    EHT1
2072  003030  075650                    EDT1
2073  003032  075740                    EFT1
2074
2075
2076                              ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2077  003034  073014                    EMT126
2078  003036  075524                    EHT1
2079  003040  075650                    EDT1
2080  003042  075740                    EFT1
2081
2082
2083                              ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
2084  C03044  073026                    EMT127
2085  003046  075524                    EHT1
2086  003050  075650                    EDT1
2087  003052  075740                    EFT1
2088
2089
2090                              ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2091  003054  073040                    EMT130
```

```
2092  003056  075524                    EMT1
2093  003060  075650                    EDT1
2094  003062  075740                    EFT1
2095
2096
2097                              ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2098  003064  073052                    EMT131
2099  003066  075524                    EMT1
2100  003070  075650                    EDT1
2101  003072  075740                    EFT1
2102
2103
2104                              ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2105  003074  073064                    EMT132
2106  003076  075524                    EMT1
2107  003100  075650                    EDT1
2108  003102  075740                    EFT1
2109
2110
2111                              ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2112  003104  073076                    EMT133
2113  003106  075524                    EMT1
2114  003110  075650                    EDT1
2115  003112  075740                    EFT1
2116
2117
2118                              ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2119  003114  073110                    EMT134
2120  003116  075524                    EMT1
2121  003120  075650                    EDT1
2122  003122  075740                    EFT1
2123
2124
2125                              ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2126  003124  073122                    EMT135
2127  003126  075524                    EMT1
2128  003130  075650                    EDT1
2129  003132  075740                    EFT1
2130
2131
2132                              ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2133  003134  073134                    EMT136
2134  003136  075524                    EMT1
2135  003140  075650                    EDT1
2136  003142  075740                    EFT1
2137
2138
2139                              ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2140  003144  073146                    EMT137
2141  003146  075524                    EMT1
2142  003150  075650                    EDT1
2143  003152  075740                    EFT1
2144
2145
2146                              ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2147  003154  073156                    EMT140
```

```
2148  003156  075524                    EMT1
2149  003160  075650                    EDT1
2150  003162  075740                    EFT1
2151
2152
2153                              ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
2154  003164  073166                    EMT141
2155  003166  075524                    EMT1
2156  003170  075650                    EDT1
2157  003172  075740                    EFT1
2158
2159
2160                              ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
2161  003174  073176                    EMT142
2162  003176  075524                    EMT1
2163  003200  075650                    EDT1
2164  003202  075740                    EFT1
2165
2166
2167                              ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
2168  003204  073206                    EMT143
2169  003206  075524                    EMT1
2170  003210  075650                    EDT1
2171  003212  075740                    EFT1
2172
2173
2174                              ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
2175  003214  073216                    EMT144
2176  003216  075524                    EMT1
2177  003220  075650                    EDT1
2178  003222  075740                    EFT1
2179
2180
2181                              ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
2182  003224  073226                    EMT145
2183  003226  075524                    EMT1
2184  003230  075650                    EDT1
2185  003232  075740                    EFT1
2186
2187
2188                              ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
2189  003234  073236                    EMT146
2190  003236  075524                    EMT1
2191  003240  075650                    EDT1
2192  003242  075740                    EFT1
2193
2194
2195                              ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
2196  003244  073246                    EMT147
2197  003246  075524                    EMT1
2198  003250  075650                    EDT1
2199  003252  075740                    EFT1
2200
2201
2202                              ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
2203  003254  073256                    EMT150
```

```
2204  003256  075524                    EMT1
2205  003260  075650                    EDT1
2206  003262  075740                    EFT1
2207
2208
2209                              ;ERROR  151     MEDIUM NOT ON LINE
2210  003264  073266                    EMT151
2211  003266  075524                    EMT1
2212  003270  075650                    EDT1
2213  003272  075740                    EFT1
2214
2215
2216                              ;ERROR  152     DRIVE FAULT
2217  003274  073300                    EMT152
2218  003276  075524                    EMT1
2219  003300  075650                    EDT1
2220  003302  075740                    EFT1
2221
2222
2223                              ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2224  003304  073312                    EMT153
2225  003306  075524                    EMT1
2226  003310  075650                    EDT1
2227  003312  075740                    EFT1
2228
2229
2230                              ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2231  003314  073330                    EMT154
2232  003316  075524                    EMT1
2233  003320  075650                    EDT1
2234  003322  075740                    EFT1
2235
2236
2237                              ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2238  003324  073346                    EMT155
2239  003326  075524                    EMT1
2240  003330  075650                    EDT1
2241  003332  075740                    EFT1
2242
2243
2244                              ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2245  003334  073360                    EMT156
2246  003336  075524                    EMT1
2247  003340  075650                    EDT1
2248  003342  075740                    EFT1
2249
2250
2251                              ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2252  003344  073372                    EMT157
2253  003346  075524                    EMT1
2254  003350  075650                    EDT1
2255  003352  075740                    EFT1
2256
2257
2258                              ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2259  003354  073404                    EMT160
```

```
2260  003356  075524                    EMT1
2261  003360  075650                    EDT1
2262  003362  075740                    EFT1
2263
2264
2265                                ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2266  003364  073414                    EMT161
2267  003366  075524                    EMT1
2268  003370  075650                    EDT1
2269  003372  075740                    EFT1
2270
2271
2272                                ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2273  003374  073426                    EMT162
2274  003376  075524                    EMT1
2275  003400  075650                    EDT1
2276  003402  075740                    EFT1
2277
2278
2279                                ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2280                                ;               WRITE BUFFER
2281  003404  075320                    EMT336
2282  003406  075606                    EMT336
2283  003410  075704                    EDT336
2284  003412  075774                    EFT336
2285
2286
2287                                ;ERROR  164     OPI SHOULD NOT BE SET
2288  003414  073450                    EMT164
2289  003416  075524                    EMT1
2290  003420  075650                    EDT1
2291  003422  075740                    EFT1
2292
2293
2294                                ;ERROR  165     IVC SHOULD NOT BE SET
2295  003424  073456                    EMT165
2296  003426  075524                    EMT1
2297  003430  075650                    EDT1
2298  003432  075740                    EFT1
2299
2300
2301                                ;ERROR  166     IAE SHOULD NOT BE SET
2302  003434  073464                    EMT166
2303  003436  075524                    EMT1
2304  003440  075650                    EDT1
2305  003442  075740                    EFT1
2306
2307
2308                                ;ERROR  167     NEM SHOULD NOT BE SET
2309  003444  073472                    EMT167
2310  003446  075524                    EMT1
2311  003450  075650                    EDT1
2312  003452  075740                    EFT1
2313
2314
2315                                ;ERROR 170      UNUSED
```

```
2316  003454  000000                    0
2317  003456  000000                    0
2318  003460  000000                    0
2319  003462  000000                    0
2320
2321
2322                                ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2323  003464  073510                    EMT171
2324  003466  075524                    EHT1
2325  003470  075650                    EDT1
2326  003472  075740                    EFT1
2327
2328
2329                                ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2330  003474  073520                    EMT172
2331  003476  075524                    EHT1
2332  003500  075650                    EDT1
2333  003502  075740                    EFT1
2334
2335
2336                                ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2337  003504  073530                    EMT173
2338  003506  075524                    EHT1
2339  003510  075650                    EDT1
2340  003512  075740                    EFT1
2341
2342
2343                                ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2344  003514  073540                    EMT174
2345  003516  075524                    EHT1
2346  003520  075650                    EDT1
2347  003522  075740                    EFT1
2348
2349
2350                                ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2351  003524  073552                    EMT175
2352  003526  075524                    EHT1
2353  003530  075650                    EDT1
2354  003532  075740                    EFT1
2355
2356
2357                                ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2358  003534  073560                    EMT176
2359  003536  075524                    EHT1
2360  003540  075650                    EDT1
2361  003542  075740                    EFT1
2362
2363
2364                                ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2365  003544  073566                    EMT177
2366  003546  075524                    EHT1
2367  003550  075650                    EDT1
2368  003552  075740                    EFT1
2369
2370
2371                                ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
 2372  003554  073600                    EMT200
 2373  003556  075524                    EHT1
 2374  003560  075650                    EDT1
 2375  003562  075740                    EFT1
 2376
 2377
 2378                          ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
 2379  003564  073612                    EMT201
 2380  003566  075524                    EHT1
 2381  003570  075650                    EDT1
 2382  003572  075740                    EFT1
 2383
 2384
 2385                          ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
 2386  003574  073624                    EMT202
 2387  003576  075524                    EHT1
 2388  003600  075650                    EDT1
 2389  003602  075740                    EFT1
 2390
 2391
 2392                          ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
 2393  003604  073636                    EMT203
 2394  003606  075524                    EHT1
 2395  003610  075650                    EDT1
 2396  003612  075740                    EFT1
 2397
 2398
 2399                          ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
 2400  003614  073650                    EMT204
 2401  003616  075524                    EHT1
 2402  003620  075650                    EDT1
 2403  003622  075740                    EFT1
 2404
 2405
 2406                          ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
 2407  003624  073666                    EMT205
 2408  003626  075524                    EHT1
 2409  003630  075650                    EDT1
 2410  003632  075740                    EFT1
 2411
 2412
 2413                          ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
 2414  003634  073676                    EMT206
 2415  003636  075524                    EHT1
 2416  003640  075650                    EDT1
 2417  003642  075740                    EFT1
 2418
 2419
 2420                          ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
 2421  003644  073706                    EMT207
 2422  003646  075524                    EHT1
 2423  003650  075650                    EDT1
 2424  003652  075740                    EFT1
 2425
 2426
 2427                          ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2428  003654  073720                    EMT210
2429  003656  075524                    EHT1
2430  003660  075650                    EDT1
2431  003662  075740                    EFT1
2432
2433
2434                                ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" - 1
2435  003664  073726                    EMT211
2436  003666  075524                    EHT1
2437  003670  075650                    EDT1
2438  003672  075740                    EFT1
2439
2440
2441                                ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2442  003674  073742                    EMT212
2443  003676  075524                    EHT1
2444  003700  075650                    EDT1
2445  003702  075740                    EFT1
2446
2447
2448                                ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2449  003704  073756                    EMT213
2450  003706  075524                    EHT1
2451  003710  075650                    EDT1
2452  003712  075740                    EFT1
2453
2454
2455                                ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2456  003714  073766                    EMT214
2457  003716  075536                    EHT2
2458  003720  075660                    EDT2
2459  003722  075750                    EFT2
2460
2461
2462                                ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2463  003724  074006                    EMT215
2464  003726  075536                    EHT2
2465  003730  075660                    EDT2
2466  003732  075750                    EFT2
2467
2468
2469                                ;ERROR  216     INCORRECT "IVC" STATUS
2470  003734  074020                    EMT216
2471  003736  075524                    EHT1
2472  003740  075650                    EDT1
2473  003742  075740                    EFT1
2474
2475
2476                                ;ERROR  217     INCORRECT "IAE" STATUS
2477  003744  074030                    EMT217
2478  003746  075524                    EHT1
2479  003750  075650                    EDT1
2480  003752  075740                    EFT1
2481
2482
2483                                ;ERROR 220      INCORRECT "WLE" STATUS
```

```
2484  003754  074040                    EMT220
2485  003756  075524                    EHT1
2486  003760  075650                    EDT1
2487  003762  075740                    EFT1
2488
2489
2490                              ;ERROR  221     INCORRECT "OPI" STATUS
2491  003764  074050                    EMT221
2492  003766  075524                    EHT1
2493  003770  075650                    EDT1
2494  003772  075740                    EFT1
2495
2496
2497                              ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2498  003774  074060                    EMT222
2499  003776  075524                    EHT1
2500  004000  075650                    EDT1
2501  004002  075740                    EFT1
2502
2503
2504                              ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2505  004004  074070                    EMT223
2506  004006  075562                    EHT223
2507  004010  075674                    EDT223
2508  004012  075764                    EFT223
2509
2510
2511                              ;ERROR  224     UNUSED
2512  004014  000000                    0
2513  004016  000000                    0
2514  004020  000000                    0
2515  004022  000000                    0
2516
2517
2518                              ;ERROR  225     UNUSED
2519  004024  000000                    0
2520  004026  000000                    0
2521  004030  000000                    0
2522  004032  000000                    0
2523
2524
2525                              ;ERROR  226     UNUSED
2526  004034  000000                    0
2527  004036  000000                    0
2528  004040  000000                    0
2529  004042  000000                    0
2530
2531
2532                              ;ERROR  227     UNUSED
2533  004044  000000                    0
2534  004046  000000                    0
2535  004050  000000                    0
2536  004052  000000                    0
2537
2538
2539                              ;ERROR  230     UNUSED
```

```
 2540  004054  000000                    0
 2541  004056  000000                    0
 2542  004060  000000                    0
 2543  004062  000000                    0
 2544
 2545
 2546                              ;ERROR  231     UNUSED
 2547  004064  000000                    0
 2548  004066  000000                    0
 2549  004070  000000                    0
 2550  004072  000000                    0
 2551
 2552
 2553                              ;ERROR  232     UNUSED
 2554  004074  000000                    0
 2555  004076  000000                    0
 2556  004100  000000                    0
 2557  004102  000000                    0
 2558
 2559
 2560                              ;ERROR  233     UNUSED
 2561  004104  000000                    0
 2562  004106  000000                    0
 2563  004110  000000                    0
 2564  004112  000000                    0
 2565
 2566
 2567                              ;ERROR  234     UNUSED
 2568  004114  000000                    0
 2569  004116  000000                    0
 2570  004120  000000                    0
 2571  004122  000000                    0
 2572
 2573
 2574                              ;ERROR  235     UNUSED
 2575  004124  000000                    0
 2576  004126  000000                    0
 2577  004130  000000                    0
 2578  004132  000000                    0
 2579
 2580
 2581                              ;ERROR  236     UNUSED
 2582  004134  000000                    0
 2583  004136  000000                    0
 2584  004140  000000                    0
 2585  004142  000000                    0
 2586
 2587
 2588                              ;ERROR  237     UNUSED
 2589  004144  000000                    0
 2590  004146  000000                    0
 2591  004150  000000                    0
 2592  004152  000000                    0
 2593
 2594
 2595                              ;ERROR  240     UNUSED
```

```
2596  004154  000000                    0
2597  004156  000000                    0
2598  004160  000000                    0
2599  004162  000000                    0
2600
2601
2602                              ;ERROR  241     UNUSED
2603  004164  000000                    0
2604  004166  000000                    0
2605  004170  000000                    0
2606  004172  000000                    0
2607
2608
2609                              ;ERROR  242     UNUSED
2610  004174  000000                    0
2611  004176  000000                    0
2612  004200  000000                    0
2613  004202  000000                    0
2614
2615
2616                              ;ERROR  243     UNUSED
2617  004204  000000                    0
2618  004206  000000                    0
2619  004210  000000                    0
2620  004212  000000                    0
2621
2622
2623                              ;ERROR  244     UNUSED
2624  004214  000000                    0
2625  004216  000000                    0
2626  004220  000000                    0
2627  004222  000000                    0
2628
2629
2630                              ;ERROR  245     UNUSED
2631  004224  000000                    0
2632  004226  000000                    0
2633  004230  000000                    0
2634  004232  000000                    0
2635
2636
2637                              ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2638  004234  074144                    EMT246
2639  004236  075524                    EHT1
2640  004240  075650                    EDT1
2641  004242  075740                    EFT1
2642
2643
2644                              ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2645  004244  074154                    EMT247
2646  004246  075524                    EHT1
2647  004250  075650                    EDT1
2648  004252  075740                    EFT1
2649
2650
2651                              ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
2652  004254  074166                    EMT250
2653  004256  075524                    EHT1
2654  004260  075650                    EDT1
2655  004262  075740                    EFT1
2656
2657
2658                                ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2659  004264  074202                    EMT251
2660  004266  075536                    EHT2
2661  004270  075660                    EDT2
2662  004272  075750                    EFT2
2663
2664
2665                                ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2666  004274  074210                    EMT252
2667  004276  075536                    EHT2
2668  004300  075660                    EDT2
2669  004302  075750                    EFT2
2670
2671
2672                                ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2673  004304  074216                    EMT253
2674  004306  075524                    EHT1
2675  004310  075650                    EDT1
2676  004312  075740                    EFT1
2677
2678
2679                                ;ERROR  254     INCORRECT "ILF" STATUS
2680  004314  074234                    EMT254
2681  004316  075524                    EHT1
2682  004320  075650                    EDT1
2683  004322  075740                    EFT1
2684
2685
2686                                ;ERROR  255     INCORRECT "ATA" STATUS
2687  004324  074244                    EMT255
2688  004326  075524                    EHT1
2689  004330  075650                    EDT1
2690  004332  075740                    EFT1
2691
2692
2693                                ;ERROR  256     INCORRECT "ILR" STATUS
2694  004334  074254                    EMT256
2695  004336  075574                    EHT256
2696  004340  075674                    EDT223
2697  004342  075764                    EFT223
2698
2699
2700                                ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2701  004344  074264                    EMT257
2702  004346  075524                    EHT1
2703  004350  075650                    EDT1
2704  004352  075740                    EFT1
2705
2706
2707                                ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
2708  004354  074276                    EMT260
2709  004356  075524                    EHT1
2710  004360  075650                    EDT1
2711  004362  075740                    EFT1
2712
2713
2714                              ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2715  004364  074310                    EMT261
2716  004366  075524                    EHT1
2717  004370  075650                    EDT1
2718  004372  075740                    EFT1
2719
2720
2721                              ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2722  004374  074330                    EMT262
2723  004376  075524                    EHT1
2724  004400  075650                    EDT1
2725  004402  075740                    EFT1
2726
2727
2728                              ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2729  004404  074340                    EMT263
2730  004406  075524                    EHT1
2731  004410  075650                    EDT1
2732  004412  075740                    EFT1
2733
2734
2735                              ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2736  004414  074350                    EMT264
2737  004416  075524                    EHT1
2738  004420  075650                    EDT1
2739  004422  075740                    EFT1
2740
2741
2742                              ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2743  004424  074370                    EMT265
2744  004426  075524                    EHT1
2745  004430  075650                    EDT1
2746  004432  075740                    EFT1
2747
2748
2749                              ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2750  004434  074410                    EMT266
2751  004436  075524                    EHT1
2752  004440  075650                    EDT1
2753  004442  075740                    EFT1
2754
2755
2756                              ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2757                              ;               ADDRESS IS TOO LARGE
2758  004444  074422                    EMT267
2759  004446  075524                    EHT1
2760  004450  075650                    EDT1
2761  004452  075740                    EFT1
2762
2763
```

```
2764
2765  004454  074440                    ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
2766  004456  075524                            EMT270
2767  004460  075650                            EHT1
2768  004462  075740                            EDT1
2769                                      EFT1
2770
2771                                      ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2772                                      ;               DIDN'T DROP
2773  004464  074454                            EMT271
2774  004466  075524                            EHT1
2775  004470  075650                            EDT1
2776  004472  075740                            EFT1
2777
2778
2779                                      ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
2780  004474  074472                            EMT272
2781  004476  075524                            EHT1
2782  004500  075650                            EDT1
2783  004502  075740                            EFT1
2784
2785
2786                                      ;ERROR  273     PIP STIL SET AFTER SEARCH
2787  004504  074510                            EMT273
2788  004506  075524                            EHT1
2789  004510  075650                            EDT1
2790  004512  075740                            EFT1
2791
2792
2793                                      ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
2794                                      ;               REGISTERS BUT MXF DID NOT SET
2795  004514  074526                            EMT274
2796  004516  075524                            EHT1
2797  004520  075650                            EDT1
2798  004522  075740                            EFT1
2799
2800
2801                                      ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2802                                      ;               COMMAND STARTED
2803  004524  074544                            EMT275
2804  004526  075524                            EHT1
2805  004530  075650                            EDT1
2806  004532  075740                            EFT1
2807
2808
2809                                      ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2810                                      ;               ZERO
2811  004534  074556                            EMT276
2812  004536  075524                            EHT1
2813  004540  075650                            EDT1
2814  004542  075740                            EFT1
2815
2816
2817                                      ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2818                                      ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2819                                      ;               3) RUN TIMED OUT
```

```
2820  004544  074572                    EMT277
2821  004546  075524                    EHT1
2822  004550  075650                    EDT1
2823  004552  075740                    EFT1
2824
2825
2826                              ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2827                              ;               WAS NOT VALID
2828  004554  074610                    EMT300
2829  004556  075524                    EHT1
2830  004560  075650                    EDT1
2831  004562  075740                    EFT1
2832
2833
2834                              ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2835                              ;               IS VALID
2836  004564  074630                    EMT301
2837  004566  075524                    EHT1
2838  004570  075650                    EDT1
2839  004572  075740                    EFT1
2840
2841
2842                              ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2843  004574  074652                    EMT302
2844  004576  075524                    EHT1
2845  004600  075650                    EDT1
2846  004602  075740                    EFT1
2847
2848
2849                              ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2850  004604  074664                    EMT303
2851  004606  075524                    EHT1
2852  004610  075650                    EDT1
2853  004612  075740                    EFT1
2854
2855
2856                              ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2857  004614  074676                    EMT304
2858  004616  075524                    EHT1
2859  004620  075650                    EDT1
2860  004622  075740                    EFT1
2861
2862
2863                              ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2864  004624  074710                    EMT305
2865  004626  075524                    EHT1
2866  004630  075650                    EDT1
2867  004632  075740                    EFT1
2868
2869
2870                              ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2871  004634  074722                    EMT306
2872  004636  075524                    EHT1
2873  004640  075650                    EDT1
2874  004642  075740                    EFT1
2875
```

```
2876
2877                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2878  004644  074732                              EMT307
2879  004646  075524                              EHT1
2880  004650  075650                              EDT1
2881  004652  075740                              EFT1
2882
2883
2884                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2885  004654  074752                              EMT310
2886  004656  075524                              EHT1
2887  004660  075650                              EDT1
2888  004662  075740                              EFT1
2889
2890
2891                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2892  004664  074764                              EMT311
2893  004666  075524                              EHT1
2894  004670  075650                              EDT1
2895  004672  075740                              EFT1
2896
2897
2898                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2899  004674  074776                              EMT312
2900  004676  075524                              EHT1
2901  004700  075650                              EDT1
2902  004702  075740                              EFT1
2903
2904
2905                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2906  004704  075010                              EMT313
2907  004706  075524                              EHT1
2908  004710  075650                              EDT1
2909  004712  075740                              EFT1
2910
2911
2912                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2913  004714  075030                              EMT314
2914  004716  075524                              EHT1
2915  004720  075650                              EDT1
2916  004722  075740                              EFT1
2917
2918
2919                                    ;ERROR  315     WRITE LOCK ERROR
2920  004724  075042                              EMT315
2921  004726  075524                              EHT1
2922  004730  075650                              EDT1
2923  004732  075740                              EFT1
2924
2925
2926                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2927  004734  075054                              EMT316
2928  004736  075524                              EHT1
2929  004740  075650                              EDT1
2930  004742  075740                              EFT1
2931
```

```
2932
2933                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2934  004744  075066                              EMT317
2935  004746  075524                              EHT1
2936  004750  075650                              EDT1
2937  004752  075740                              EFT1
2938
2939
2940                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2941  004754  075076                              EMT320
2942  004756  075524                              EHT1
2943  004760  075650                              EDT1
2944  004762  075740                              EFT1
2945
2946
2947                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2948  004764  075106                              EMT321
2949  004766  075524                              EHT1
2950  004770  075650                              EDT1
2951  004772  075740                              EFT1
2952
2953
2954                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
2955  004774  075116                              EMT322
2956  004776  075524                              EHT1
2957  005000  075650                              EDT1
2958  005002  075740                              EFT1
2959
2960
2961                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
2962  005004  075124                              EMT323
2963  005006  075524                              EHT1
2964  005010  075650                              EDT1
2965  005012  075740                              EFT1
2966
2967
2968                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2969  005014  075134                              EMT324
2970  005016  075524                              EHT1
2971  005020  075650                              EDT1
2972  005022  075740                              EFT1
2973
2974
2975                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2976  005024  075146                              EMT325
2977  005026  075524                              EHT1
2978  005030  075650                              EDT1
2979  005032  075740                              EFT1
2980
2981
2982                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
2983  005034  075160                              EMT326
2984  005036  075524                              EHT1
2985  005040  075650                              EDT1
2986  005042  075740                              EFT1
2987
```

```
2988
2989                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
2990  005044  075176                              EMT327
2991  005046  075524                              EHT1
2992  005050  075650                              EDT1
2993  005052  075740                              EFT1
2994
2995
2996                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
2997  005054  075210                              EMT330
2998  005056  075524                              EHT1
2999  005060  075650                              EDT1
3000  005062  075740                              EFT1
3001
3002
3003                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3004  005064  075220                              EMT331
3005  005066  075524                              EHT1
3006  005070  075650                              EDT1
3007  005072  075740                              EFT1
3008
3009
3010                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3011  005074  075232                              EMT332
3012  005076  075524                              EHT1
3013  005100  075650                              EDT1
3014  005102  075740                              EFT1
3015
3016
3017                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3018  005104  075244                              EMT333
3019  005106  075524                              EHT1
3020  005110  075650                              EDT1
3021  005112  075740                              EFT1
3022
3023
3024                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3025  005114  075262                              EMT334
3026  005116  075524                              EHT1
3027  005120  075650                              EDT1
3028  005122  075740                              EFT1
3029
3030
3031                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3032  005124  075300                              EMT335
3033  005126  075524                              EHT1
3034  005130  075650                              EDT1
3035  005132  075740                              EFT1
3036
3037
3038                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3039  005134  075320                              EMT336
3040  005136  075606                              EHT336
3041  005140  075704                              EDT336
3042  005142  075774                              EFT336
3043
```

```
3044
3045                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3046  005144  075330                              EMT337
3047  005146  075620                              EHT337
3048  005150  075714                              EDT337
3049  005152  076004                              EFT337
3050
3051
3052                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3053  005154  075340                              EMT340
3054  005156  075606                              EHT336
3055  005160  075704                              EDT336
3056  005162  075774                              EFT336
3057
3058
3059                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3060  005164  075352                              EMT341
3061  005166  075606                              EHT336
3062  005170  075704                              EDT336
3063  005172  075774                              EFT336
3064
3065
3066                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3067  005174  075360                              EMT342
3068  005176  075524                              EHT1
3069  005200  075650                              EDT1
3070  005202  075740                              EFT1
3071
3072
3073                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3074  005204  075372                              EMT343
3075  005206  075524                              EHT1
3076  005210  075650                              EDT1
3077  005212  075740                              EFT1
3078
3079
3080                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3081  005214  075404                              EMT344
3082  005216  075632                              EHT344
3083  005220  075724                              EDT344
3084  005222  076014                              EFT344
3085
3086
3087                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3088  005224  075416                              EMT345
3089  005226  075524                              EHT1
3090  005230  075650                              EDT1
3091  005232  075740                              EFT1
3092
3093
3094                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3095  005234  075426                              EMT346
3096  005236  075524                              EHT1
3097  005240  075650                              EDT1
3098  005242  075740                              EFT1
3099
```

```
3100
3101                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3102  005244  075442                             EMT347
3103  005246  075524                             EHT1
3104  005250  075650                             EDT1
3105  005252  075740                             EFT1
3106
3107
3108                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3109  005254  075454                             EMT350
3110  005256  075524                             EHT1
3111  005260  075650                             EDT1
3112  005262  075740                             EFT1
3113
3114
3115                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3116  005264  075472                             EMT351
3117  005266  075524                             EHT1
3118  005270  075650                             EDT1
3119  005272  075740                             EFT1
3120
3121
3122                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3123  005274  075502                             EMT352
3124  005276  000000                             0
3125  005300  000000                             0
3126  005302  000000                             0
3127
3128
3129                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3130  005304  075506                             EMT353
3131  005306  075644                             EHT353
3132  005310  075736                             EDT353
3133  005312  076024                             EFT353
3134
3135
3136                                    ;ERROR  354     BSE SHOULD NOT BE SET
3137  005314  075516                             EMT354
3138  005316  075524                             EHT1
3139  005320  075650                             EDT1
3140  005322  075740                             EFT1
3141
3142
3143                                    ;PUT ERROR TABLE HERE
3144                                    .SBTTL  ERROR TABLE USAGE
3145
3146                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3147                                    ;NUMBER, I.E.,
3148                                    ;
3149                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3150                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3151                                    ;       EDT - ERROR DATA TABLE ADDRESS
3152                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3153                                    ;
3154                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3155                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3156                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3157                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3158                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3159                    ;
3160                    ;SIMILARLY, THE EMT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3161                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3162                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3163                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3164                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3165                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3166                    ;MUST ALSO HAVE A FORMAT.
3167                    ;
3168                    ;IN SUMMARY,
3169                    ;
3170                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3171                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3172                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3173                    ;
3174
```

```
                                        .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     #SWR,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  064576  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005370  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005376  012737  065172  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005404  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005412  012737  066732  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005420  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
005426  012737  067040  000024                  MOV     #$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     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005612                                  68$:
```

```
3231                                        .SBTTL  TYPE PROGRAM NAME
3232                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
3233  005612  005227  177777                        INC     #-1             ;;FIRST TIME?
3234  005616  001052                              BNE     69$             ;;BRANCH IF NO
3235  005620  022737  037176  000042              CMP     #$ENDAD,@#42    ;;AC'T-11?
3236  005626  001446                              BEQ     69$             ;;BRANCH IF YES
3237  005630  104401  005676                      TYPE    ,70$            ;;TYPE ASCIZ STRING
3238                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3239  005634  005737  000042                      TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3240  005640  001012                              BNE     71$             ;;BRANCH IF YES
3241  005642  123727  001242  000001              CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
3242  005650  001406                              BEQ     71$             ;;BRANCH IF YES
3243  005652  023727  001154  000176              CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3244  005660  001005                              BNE     72$             ;;BRANCH IF NO
3245  005662  104407                              GTSWR                   ;;GET SOFT-SWR SETTINGS
3246  005664  000403                              BR      72$
3247  005666  112737  000001  001150  71$:       MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3248  005674                        72$:
3249  005674  000423                              BR      69$             ;;GET OVER THE ASCIZ
3250                                  ;;70$:      .ASCIZ  <CRLF>@RM03/RM02 FUNCTIONAL TEST, PART 3 @<CRLF>
3251  005744                        69$:
3252
3253
3254                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
3255  005744  122737  000077  000041              CMPB    #77,@#41        ;LOAD FROM THE RM03 FOR XXDP
3256  005752  001003                              BNE     5$              ;BRANCH IF NOT
3257  005754  104401  070563                      TYPE    ,XDPMG          ;THPE CHANGE PACK MESSAGE
3258  005760  000000                              HALT                    ;THEN HALT
3259  005762                        5$:
3260  005762  005737  000042                      TST     42              ;IS LOC 42 ZERO ??
3261  005766  001003                              BNE     10$             ;NO - NOT IN STANDALONE
3262  005770  105737  001242                      TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
3263  005774  001451                              BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
3264  005776                        .0$:
3265
3266                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
3267  005776  132737  000200  001243  XSIZ:       BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
3268  006004  001043                              BNE     20$             ;NO
3269                                  ;           MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
3270  006006  005037  001300                      CLR     $DEVM           ;CLEAR THE BIT MAP
3271  006012  005001                              CLR     R1              ;START FROM THE DRIVE 0
3272  006014  012704  000001                      MOV     #BIT0,R4        ;BIT MAP FOR DRIVE 0
3273  006020  013700  001276                      MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
3274  006024  012760  000040  000010  15$:       MOV     #CLR,RMCS2(R0)  ;MASS BUS CLEAR
3275  006032  010160  000010                      MOV     R1,RMCS2(R0)    ;LOAD THE BDRIVE ADDRESS
3276  006036  016003  000010                      MOV     RMCS2(R0),R3    ;READ STATUS REG TO SEE NED BIT
3277  006042  032703  010000                      BIT     #NED,R3         ;NED BIT SET ?
3278  006046  001010                              BNE     16$             ;BRANCH IF SO
3279  006050  016003  000000                      MOV     RMCS1(R0),R3    ;READ DVA BIT
3280  006054  032703  004000                      BIT     #DVA,R3         ;DVA SET
3281  006060  001403                              BEQ     16$             ;BRANCH IF NOT
3282  006062  050437  001300                      BIS     R4,$DEVM        ;SET THE DEVICE MAP
3283  006066  000405                              BR      17$             ;NOT TYPE THE NON-EXIST MESG
3284  006070                        16$:
3285  006070  104401  070650                      TYPE    ,NOTEX          ;TYPE NOT EXIST DRIVE
3286  006074  010146                              MOV     R1,-(SP)        ;DRIVE NUMBER
```

```
3287  006076  104403                  TYPOS
3288  006100     006                  .BYTE   6
3289  006101     000                  .BYTE   0
3290  006102  005201           17$:    INC     R1              ;INCREMENT THE DRIVE ADDRESS
3291  006104  006304                   ASL     R4              ;SET UP BIT MAP FOR THE NEXT DRIVE
3292  006106  022701  000007           CMP     #7,R1           ;ALL DRIVES ARE CHECKED /
3293  006112  103344                   BHIS    15$             ;BRANCH IF NOT
3294  006114                   20$:
3295
3296                           ;GO TO COMMON START CODE
3297  006114  000137  006746           JMP     CMNSTART
```

```
3298  006120                          STANDALONE:
3299  006120  004737  065402                JSR     PC,$TKINT       ;INITIALIZE CONSOLE
3300
3301                                  ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3302  006124  005327  000001                DEC     #1              ;FIRST START ??
3303  006130  100024                        BPL     10$             ;YES !!
3304
3305
3306                                  ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3307  006132  104401  070072        5$:     TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
3308  006136  104411                        RDCHR                   ;GET RESPONSE
3309  006140  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3310  006144  104401  001176                TYPE    ,$TMP1
3311  006150  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE??
3312  006156  001002                        BNE     6$              ;NO!!
3313  006160  000137  007100                JMP     READY           ;KEEP PREVIOUS PARAMETERS
3314  006164  123727  001176  000116  6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
3315  006172  001420                        BEQ     20$             ;GET NEW PARAMETERS
3316  006174  104401  067603                TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
3317  006200  000754                        BR      5$              ;RETRY
3318  006202                          10$:
3319
3320                                  ;SEE IF OPERATOR WANTS HELP FILE
3321  006202  104401  067605                TYPE    ,HELPQST        ;WANT HELP ??
3322  006206  104411                        RDCHR                   ;GET RESPONSE
3323  006210  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3324  006214  104401  001176                TYPE    ,$TMP1
3325  006220  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3326  006226  001002                        BNE     20$             ;NO - DONT TYPE HELP
3327  006230  104401  107546                TYPE    ,HELP           ;YES - TYPE HELP TEXT
3328  006234                          20$:
3329
3330                                  ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3331  006234  104401  067773                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
3332  006240  104411                        RDCHR                   ;GET RESPONSE
3333  006242  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3334  006246  104401  001176                TYPE    ,$TMP1
3335  006252  104411                        RDCHR                   ;READ ONE MORE CHARACTER
3336  006254  005726                        TST     (SP)+           ;CLEAR THE STACK POINT
3337  006256  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3338  006264  001137                        BNE     90$             ;NO !!
3339  006266                          30$:
3340
3341                                  ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3342  006266  104401  070131                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
3343  006272  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
3344  006276  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3345  006300  104401  067574                TYPE    ,CLSPRN
3346  006304  104413                        RDOCT                   ;GET NEW BUS ADDRESS
3347  006306  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3348  006312  001412                        BEQ     50$             ;YES-SKIP TO NEXT ENTRY
3349  006314  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
3350  006322  101403                        BLOS    40$             ;YES
3351  006324  104401  070156                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
3352  006330  000756                        BR      30$             ;RETRY
3353  006332  013737  001176  001276  40$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
```

```
3354  006340  113737  001272  001176  50$:    MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
3355  006346  105037  001177                  CLRB    $TMP1+1
3356  006352  104401  070237                  TYPE    ,CNSL03
3357  006356  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3358  006362  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3359  006364     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3360  006365     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3361  006366  104401  067574                  TYPE    ,CLSPRN
3362  006372  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
3363  006374  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
3364  006400  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
3365  006402  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
3366  006410  101003                          BHI     60$             ;YES!!
3367  006412  104401  070267                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
3368  006416  000750                          BR      50$             ;RETRY
3369  006420  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
3370  006426  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
3371  006434  006237  001176                  ASR     $TMP1
3372  006440  006237  001176                  ASR     $TMP1
3373  006444  006237  001176                  ASR     $TMP1
3374  006450  006237  001176                  ASR     $TMP1
3375  006454  006237  001176                  ASR     $TMP1
3376  006460  105037  001177                  CLRB    $TMP1+1
3377  006464  104401  070343                  TYPE    ,CNSL05
3378  006470  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3379  006474  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3380  006476     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3381  006477     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3382  006500  104401  067574                  TYPE    ,CLSPRN
3383  006504  104413                          RDOCT                   ;GET NEW PRIORITY
3384  006506  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3385  006512  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
3386  006514  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
3387  006522  002403                          BLT     80$             ;YES!!
3388  006524  104401  070377                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
3389  006530  000736                          BR      70$             ;RETRY
3390  006532                          80$:                            ;STORE NEW PRIORITY
3391  006532  006337  001176                  ASL     $TMP1
3392  006536  006337  001176                  ASL     $TMP1
3393  006542  006337  001176                  ASL     $TMP1
3394  006546  006337  001176                  ASL     $TMP1
3395  006552  006337  001176                  ASL     $TMP1
3396  006556  113737  001176  001273          MOVB    $TMP1,$VECT1+1
3397  006564                          90$:
3398
3399                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3400  006564  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
3401  006570  104401  070424                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
3402  006574  104401  067577                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3403  006600  104411                          RDCHR                   ;GET RESPONSE
3404  006602  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3405  006606  104401  001176                  TYPE    ,$TMP1
3406  006612  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
3407  006620  001021                          BNE     110$            ;NO
3408  006622  000137  005776                  JMP     XSIZ            ;AUTO SIZE
3409  006626  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
```

```
 3410  006634  000444                          BR      140$            ;SKIP TO NEXT ENTRY
 3411  006636  104401  067577          100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
 3412  006642  104411                          RDCHR                   ;GET RESPONSE
 3413  006644  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
 3414  006650  104401  001176                  TYPE    ,$TMP1
 3415  006654  023727  001176  000015          CMP     $TMP1,#CR       ;CARRIAGE RETURN??
 3416  006662  001431                          BEQ     140$
 3417  006664  023727  001176  000060  110$:   CMP     $TMP1,#'0       ;NUMBER < 0??
 3418  006672  002404                          BLT     120$            ;YES!!
 3419  006674  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7??
 3420  006702  003403                          BLE     130$            ;NO!!
 3421  006704  104401  067603          120$:   TYPE    ,QSTMRK         ;TYPE "?"
 3422  006710  000752                          BR      100$            ;RETRY
 3423  006712  013701  001176          130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
 3424  006716  042701  177770                  BIC     #^C7,R1
 3425  006722  116102  070774                  MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
 3426  006726  042702  177400                  BIC     #^C377,R2               ;CLEAR UNUSED BITS
 3427  006732  050237  001300                  BIS     R2,$DEVM                ;SET DEVICE # IN MAP
 3428  006736  122737  000377  001300          CMPB    #377,$DEVM      ;DONE ??
 3429  006744  101334                          BHI     100$            ;NO
 3430  006746                          140$:
 3431
```

```
3432  006746                                CMNCTART:
3433
3434                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
3435  006746  013700  001300                        MOV     $DEVM,R0        ;R0 = DEVICE MAP
3436  006752  012701  001452                        MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
3437  006756  010137  001450                        MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
3438  006762  012702  000001                        MOV     #1,R2           ;R2 = DEVICE POINTER
3439  006766  005003                              CLR     R3              ;R3 = DEVICE NUMBER
3440  006770  030200                      10$:     BIT     R2,R0           ;IS THIS DEVICE IN MAP ??
3441  006772  001406                              BEQ     20$             ;NO !!
3442  006774  010311                              MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
3443  006776  116361  070774  000001              MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
3444  007004  062701  000002                      ADD     #2,R1           ;ADVANCE ENTRY POINTER
3445  007010  006302                      20$:     ASL     R2              ;ADVANCE DEVICE POINTER
3446  007012  105702                              TSTB    R2              ;DONE ALL DEVICES ??
3447  007014  001402                              BEQ     25$             ;YES
3448  007016  005203                              INC     R3              ;ADVANCE DEVICE NUMBER
3449  007020  000763                              BR      10$             ;ENTER NEXT DEVICE
3450  007022  005011                      25$:     CLR     (R1)            ;TERMINATE TEST QUE
3451
3452                                        ;SIZE FOR CLOCK
3453  007024  004737  044224                        JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
3454  007030  000403                              BR      40$             ;YES - CLOCK IS PRESENT
3455  007032  104000                      30$:     ERROR                   ;NO CLOCK
3456  007034  000000                              HALT
3457  007036  000775                              BR      30$
3458  007040                              40$:
3459                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3460  007040  005737  000042                      TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3461  007044  001012                              BNE     64$             ;;BRANCH IF YES
3462  007046  123727  001242  000001              CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
3463  007054  001406                              BEQ     64$             ;;BRANCH IF YES
3464  007056  023727  001154  000176              CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3465  007064  001005                              BNE     65$             ;;BRANCH IF NO
3466  007066  104407                              GTSWR                   ;;GET SOFT-SWR SETTINGS
3467  007070  000403                              BR      65$
3468  007072  112737  000001  001150      64$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3469  007100                              65$:
3470  007100  000240                      READY:   NOP                     ;READY TO START TEST
3471  007102  005037  001326                      CLR     CTLFG           ;CLEAR THE CONTROL-C FLAG
3472  007106  004737  065402                      JSR     PC,$TKINT       ;INITIALIZE TTY
3473  007112  013746  000300                      MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
3474  007116  012746  007124                      MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3475  007122  000002                              RTI                     ;;POP NEW PC AND PS
3476  007124                              64$:
3477  007124  117737  172320  001234              MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
3478  007132  005037  001474                      CLR     MEDENB          ;CLEAR MEDIA ENABLE
```

```
3479                                    ;;*****************************************************************
3480                                    ;*TEST 1         CONTROLLER ACCESS TEST
3481                                    ;;*****************************************************************
3482  007136                            TST1:
3483  007136  000240                           NOP                     ;START OF TEST
3484  007140  012737  007154  001122           MOV     #1$,$LPADR
3485  007146  012737  007154  001124           MOV     #1$,$LPERR
3486  007154                            1$:
3487  007154  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3488  007160  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3489  007164  013701  001450                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3490  007170  012737  000001  001226           MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3491  007176  005001                           CLR     R1
3492  007200  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
3493  007204  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
3494  007210  012737  007302  000004           MOV     #3$,ERRVEC
3495  007216  012737  000300  000006           MOV     #PR6,ERRVEC+2
3496
3497  007224  110160  000001                   MOVB    R1,RMCS1+1(R0)  ;MOVE HI BYTE TO RMCS1
3498  007230  010160  000002                   MOV     R1,RMWC(R0)     ;MOVE WORD COUNT REGISTER
3499  007234  016002  000002                   MOV     RMWC(R0),R2
3500  007240  010160  000004                   MOV     R1,RMBA(R0)     ;MOVE BUS ADDRESS REGISTER
3501  007244  016002  000004                   MOV     RMBA(R0),R2
3502  007250  010160  000010                   MOV     R1,RMCS2(R0)    ;MOVE CONTROL STATUS REGISTER
3503  007254  016002  000010                   MOV     RMCS2(R0),R2
3504  007260  010160  000022                   MOV     R1,RMDB(R0)     ;MOVE DATA BUFFER
3505  007264  016002  000022                   MOV     RMDB(R0),R2
3506  007270  012637  000006                   MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3507  007274  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3508  007300  000415                           BR      7$              ;NO BUS TIMEOUT OCCURRED
3509
3510  007302  022626                    3$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
3511  007304  012637  000006                   MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3512  007310  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3513  007314  104110                           ERROR   110             ;CANNOT ACCESS MASSBUS CONTROLLER
3514  007316  005737  000042                   TST     42              ;STAND ALONE MODE??
3515  007322  001002                           BNE     5$              ;NO!!
3516  007324  000137  005324                   JMP     START           ;YES-GO GET $BASE
3517  007330  000137  037006            5$:    JMP     $EOP            ;GO TO END OF PASS HANDLER
3518  007334                            7$:
3519
3520                                    ;;*****************************************************************
3521                                    ;*TEST 2         DEVICE AVAILABLE TEST
3522
3523                                    ;;*****************************************************************
3524  007334                            TST2:
3525  007334  000004                           SCOPE                   ;SCOPE CALL
3526  007336  000240                           NOP                     ;START OF TEST
3527  007340  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3528  007344  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3529  007350  013701  001450                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3530  007354  012737  000002  001226           MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3531
3532  007362  004737  052700                   JSR     PC,CNTCLR
3533  007366  000404                           BR      2$      ;GO TO 2$ IF NO ERROR
3534  007370  000240                           NOP                     ;RETURN HERE IF ERROR
```

```
3535  007372  104000                        ERROR                       ;ERROR NUMBER DEFINED BY SUB
3536  007374  000137  007514                JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
3537  007400                        2$:
3538  007400  013746  000004                MOV     ERRVEC,-(SP)        ;;PUSH ERRVEC ON STACK
3539  007404  013746  000006                MOV     ERRVEC+2,-(SP)              ;;PUSH ERRVEC+2 ON STACK
3540  007410  012737  007500  000004        MOV     #5$,ERRVEC
3541  007416  013737  000300  000006        MOV     PR6,ERRVEC+2
3542
3543  007424  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
3544  007432  016037  000010  001174        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
3545  007440  012637  000006                MOV     (SP)+,ERRVEC+2              ;;POP STACK INTO ERRVEC+2
3546  007444  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3547  007450  032737  010000  001174        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
3548  007456  001402                        BEQ     3$              ;NO!!
3549  007460  104111                        ERROR   111             ;NONEXISTENT DEVICE
3550  007462  000414                        BR      7$
3551  007464  032737  004000  001176  3$:   BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
3552  007472  001012                        BNE     9$              ;YES!!
3553  007474  104112                        ERROR   112             ;DEVICE NOT AVAILABLE
3554  007476  000406                        BR      7$
3555
3556  007500  022626                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
3557  007502  012637  000006                MOV     (SP)+,ERRVEC+2              ;;POP STACK INTO ERRVEC+2
3558  007506  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3559  007512  104113                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
3560  007514  000137  036752        7$:     JMP     $EOSP
3561
3562  007520                        9$:
3563
3564                                ;;**************************************************************
3565                                ;*TEST 3         DRIVE TYPE TEST
3566
3567                                ;;**************************************************************
3568  007520                        TST3:
3569  007520  000004                        SCOPE                       ;SCOPE CALL
3570  007522  000240                        NOP                         ;START OF TEST
3571  007524  012706  001100                MOV     #STACK,SP           ;INITIALIZE STACK POINTER
3572  007530  013700  001276                MOV     $BASE,R0            ;R0=UNIBUS ADDRESS
3573  007534  013701  001450                MOV     TSTQUE,R1           ;(R1) = DEVICE BEING TESTED
3574  007540  012737  000003  001226        MOV     #3,$TESTN           ;;SET TEST NUMBER IN APT MAIL BOX
3575
3576  007546  004737  052700                JSR     PC,CNTCLR
3577  007552  000404                        BR      2$      ;GO TO 2$ IF NO ERROR
3578  007554  000240                        NOP                         ;RETURN HERE IF ERROR
3579  007556  104000                        ERROR                       ;ERROR NUMBER DEFINED BY SUB
3580  007560  000137  007676                JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
3581  007564                        2$:
3582  007564  112737  000026  001506        MOVB    #RMDT,GETINX        ;SETUP GET INDEX TABLE
3583  007572  112737  000200  001507        MOVB    #200,GETINX+1
3584  007600  012737  007702  001356        MOV     #5$,RMDTI           ;RMDT INPUT BUFFER = 5$
3585  007606  004737  043536                JSR     PC,GET              ;GO GET DRIVE TYPE
3586  007612  000402                        BR      3$      ;GO TO 3$ IF NO ERROR
3587  007614  000240                        NOP                         ;RETURN HERE IF ERROR
3588  007616  104000                        ERROR                       ;ERROR NUMBER SPECIFIED BY GET
3589  007620  022737  020024  001356  3$:   CMP     #SNGPRT,RMDTI       ;SINGLE PORT RM03??
3590  007626  001425                        BEQ     5$                  ;YES!!
```

```
3591  007630  022737  024024  001356          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3592  007636  001421                          BEQ     5$              ;YES!!
3593  007640  022737  020025  001356          CMP     #SNGPRT!BIT0,RMDTI      ;SINGLE PROT RM02 ?
3594  007646  001415                          BEQ     5$              ;
3595  007650  022737  024025  001356          CMP     #DULPRT!BIT0,RMDTI      ;DUAL PORT RM02 ?
3596  007656  001411                          BEQ     5$              ;
3597  007660  012737  020024  001176          MOV     #SNGPRT,$TMP1
3598  007666  012737  024024  001200          MOV     #DULPRT,$TMP2
3599  007674  104114                          ERROR   114             ;NOT AN RM03
3600  007676  000137  036752          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3601
3602  007702                          5$:
3603                                  ;;*****************************************************************
3604                                  ;*TEST 4        WRITE, READ ZEROS
3605                                  ;;*****************************************************************
3606  007702                          TST4:
3607  007702  000004                          SCOPE                   ;SCOPE CALL
3608  007704  000240                          NOP                     ;START OF TEST
3609  007706  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3610  007712  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3611  007716  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3612  007722  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3613
3614                                  ;;*****************************************************************
3615                                  ;LOOP #1        FORMAT,WRITE,READ
3616
3617  007730                          10$:
3618
3619                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
3620  007730  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3621  007734  054130                          .WORD   054130  ;TASK DESCRIPTOR
3622  007736  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
3623  007740  000240                          NOP                     ;RETURN HERE IF ERROR
3624  007742  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3625  007744  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3626  007750                          20$:
3627
3628                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
3629  007750  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
3630  007756  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
3631  007764  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
3632  007772  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
3633  010000  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
3634  010006  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
3635
3636                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
3637
3638  010014  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
3639  010020  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
3640  010022  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
3641  010026  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
3642  010030  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3643  010034                          25$:
3644  010034  012737  071106  001174          MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
3645  010042  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
3646  010050  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
```

```
3647
3648                                    ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
3649  010054  012702  001535                      MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
3650  010060  112722  000034                      MOVB    #RMDC,(R2)+
3651  010064  112722  000006                      MOVB    #RMDA,(R2)+
3652  010070  112722  000004                      MOVB    #RMBA,(R2)+
3653  010074  112722  000002                      MOVB    #RMWC,(R2)+
3654  010100  112722  000032                      MOVB    #RMOF,(R2)+
3655  010104  112722  000000                      MOVB    #RMCS1,(R2)+
3656  010110  112712  000200                      MOVB    #200,(R2)               ;TERMINATE TABLE
3657  010114                                30$:
3658
3659                                    ;FORMAT THE DRIVE
3660  010114  004737  044006                      JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3661  010120  000404                               BR      40$                     ;GO TO 40$ IF NO ERROR
3662  010122  000240                               NOP                     ;RETURN HERE IF ERROR
3663  010124  104000                               ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3664  010126  000137  010724                       JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3665  010132                                40$:
3666
3667                                    ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
3668  010132  004737  043452                      JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
3669
3670                                    ;WAIT FOR THE FORMAT TO COMPLETE
3671  010136  004737  044346                      JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
3672
3673                                    ;GO READ STATUS FOR FORMAT OPERATION
3674  010142  004737  043536                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3675  010146  000404                               BR      50$                     ;GO TO 50$ IF NO ERROR
3676  010150  000240                               NOP                     ;RETURN HERE IF ERROR
3677  010152  104000                               ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3678  010154  000137  010724                       JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3679  010160                                50$:
3680
3681                                    ;VERIFY NO ERRORS DURING FORMAT
3682  010160  004737  057036                      JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3683  010164  000405                               BR      60$                     ;GO TO 60$ IF NO ERROR
3684  010166  000240                               NOP                     ;RETURN HERE IF ERROR
3685  010170  104000                               ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3686  010172  004736                               JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3687  010174  000137  010724                       JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3688  010200                                60$:
3689
3690                                    ;MOVE LOOP ADDRESSES TO NEXT OPERATION
3691  010200  012737  010216  001122               MOV     #70$,$LPADR
3692  010206  012737  010216  001124               MOV     #70$,$LPERR
3693  010214  000410                               BR      80$                     ;SKIP TO WRITE OPERATION
3694
3695                                    ;;*****************************************************************
3696                                    ;LOOP #2        WRITE,READ
3697
3698  010216                                70$:
3699
3700
3701                                    ;PREPARE DEVICE FOR WRITE OPERATION
3702  010216  004737  040020                      JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
```

```
3703  010222  054130                        .WORD   054130  ;TASK DESCRIPTOR
3704  010224  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
3705  010226  000240                        NOP                     ;RETURN HERE IF ERROR
3706  010230  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3707  010232  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3708  010236                        80$:
3709
3710
3711  010236                        120$:
3712
3713                                ;WRITE DATA TO THE DRIVE
3714  010236  012737  107552  001404        MOV     #BUFONE+4,RMBAO         ;MOVE MEMORY ADDRESS
3715  010244  012737  177400  001402        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
3716  010252  012737  000061  001400        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
3717  010260  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3718  010264  112722  000006                MOVB    #RMDA,(R2)+
3719  010270  112722  000034                MOVB    #RMDC,(R2)+
3720  010274  112722  000032                MOVB    #RMOF,(R2)+
3721  010300  112722  000004                MOVB    #RMBA,(R2)+
3722  010304  112722  000002                MOVB    #RMWC,(R2)+
3723  010310  112722  000000                MOVB    #RMCS1,(R2)+
3724  010314  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
3725
3726  010320  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3727  010324  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
3728  010326  000240                        NOP                     ;RETURN HERE IF ERROR
3729  010330  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3730  010332  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3731  010336                        130$:
3732
3733                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3734  010336  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3735
3736                                ;WAIT FOR WRITE COMMAND TO COMPLETE
3737  010342  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3738
3739                                ;GO READ STATUS FOR WRITE COMMAND
3740  010346  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3741  010352  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
3742  010354  000240                        NOP                     ;RETURN HERE IF ERROR
3743  010356  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3744  010360  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3745  010364                        140$:
3746
3747                                ;CHECK FOR ERRORS DURING WRITE OPERATION
3748  010364  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3749  010370  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
3750  010372  000240                        NOP                     ;RETURN HERE IF ERROR
3751  010374  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3752  010376  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3753  010400  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3754  010404                        150$:
3755  010404  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3756  010410  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
3757  010412  000240                        NOP                     ;RETURN HERE IF ERROR
3758  010414  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
```

```
                                T4      WRITE, READ ZEROS

3759  010416  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3760  010420  000137  010724            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3761  010424                        160$:
3762
3763  010424                        170$:
3764  010424  004737  045364            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3765  010430  000405                    BR      180$                    ;GO TO 180$ IF NO ERROR
3766  010432  000240                    NOP                     ;RETURN HERE IF ERROR
3767  010434  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3768  010436  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3769  010440  000137  010724            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3770  010444                        180$:
3771
3772                                ;CHANGE LOOP ADDRESSES
3773  010444  012737  010462  001122    MOV     #190$,$LPADR
3774  010452  012737  010462  001124    MOV     #190$,$LPERR
3775  010460  000410                    BR      200$                    ;SKIP TO NEXT OPERATION
3776
3777                                ;;**************************************************************
3778                                ;LOOP #3        READ
3779
3780  010462                        190$:
3781
3782                                ;PREPARE DEVICE FOR READ OPERATION
3783  010462  004737  040020            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3784  010466  054130                    .WORD   054130  ;TASK DESCRIPTOR
3785  010470  000404                    BR      200$                    ;GO TO 200$ IF NO ERROR
3786  010472  000240                    NOP                     ;RETURN HERE IF ERROR
3787  010474  104000                    ERROP                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3788  010476  000137  010724            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3789  010502                        200$:
3790
3791  010502                        240$:
3792
3793                                ;READ DATA FROM DEVICE
3794  010502  012737  110556  001404    MOV     #BUFTWO+4,RMBAO         ;CHANGE MEMORY ADDRESS
3795  010510  012737  000071  001400    MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
3796  010516  012702  001535            MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3797  010522  112722  000006            MOVB    #RMDA,(R2)+
3798  010526  112722  000032            MOVB    #RMOF,(R2)+
3799  010532  112722  000034            MOVB    #RMDC,(R2)+
3800  010536  112722  000004            MOVB    #RMBA,(R2)+
3801  010542  112722  000002            MOVB    #RMWC,(R2)+
3802  010546  112722  000000            MOVB    #RMCS1,(R2)+
3803  010552  112712  000200            MOVB    #200,(R2)
3804
3805  010556  004737  044006            JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3806  010562  000404                    BR      250$                    ;GO TO 250$ IF NO ERROR
3807  010564  000240                    NOP                     ;RETURN HERE IF ERROR
3808  010566  104000                    ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3809  010570  000137  010724            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3810  010574                        250$:
3811
3812                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3813  010574  004737  043452            JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3814
```

```
3815
3816  010600  004737  044346                ;WAIT FOR READ OPERATION TO COMPLETE
3817                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3818
3819  010604  004737  043536                ;GO READ STATUS FOR READ OPERATION
3820  010610  000404                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3821  010612  000240                        BR      260$                    ;GO TO 260$ IF NO ERROR
3822  010614  104000                        NOP                     ;RETURN HERE IF ERROR
3823  010616  000137  010724                ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3824  010622                                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3825                                      260$:
3826                                      ;CHECK FOR ERRORS DURING READ OPERATION
3827  010622  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3828  010626  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
3829  010630  000240                        NOP                     ;RETURN HERE IF ERROR
3830  010632  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3831  010634  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3832  010636  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3833  010642                                270$:
3834  010642  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3835  010646  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
3836  010650  000240                        NOP                     ;RETURN HERE IF ERROR
3837  010652  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3838  010654  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3839  010656  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3840  010662                                280$:
3841
3842  010662                                290$:
3843  010662  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3844  010666  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
3845  010670  000240                        NOP                     ;RETURN HERE IF ERROR
3846  010672  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3847  010674  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3848  010676  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3849  010702                                300$:
3850  010702  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
3851  010706  107552                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
3852  010710  110556                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
3853  010712  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
3854  010714  000240                        NOP                     ;RETURN HERE IF ERROR
3855  010716  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3856  010720  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3857  010724                                310$:
3858
3859  010724                                350$:
3860                                        ;;*****************************************************************
3861                                        ;*TEST 5         WRITE, WRITE CHECK ZEROS
3862                                        ;;*****************************************************************
3863  010724                                TST5:
3864  010724  000004                        SCOPE                   ;SCOPE CALL
3865  010726  000240                        NOP                     ;START OF TEST
3866  010730  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3867  010734  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3868  010740  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3869  010744  012737  000005  001226        MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3870
```

```
                                        ;;*****************************************************************
                                        ;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,RMDCO                ;CYLINDER = 0
011000  012737  000000  001406                MOV     #0,RMDAO                ;TRACK = 0, SECTOR = 0
011006  012737  107546  001404                MOV     #BUFONE,RMBAO           ;BUS ADDRESS
011014  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWCO           ;WORD COUNT = 1 SECTOR
011022  012737  010000  001432                MOV     #FMT16,RMOFO            ;16 BIT FORMAT
011030  012737  000063  001400                MOV     #WH!GO,RMCS1O           ;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  067464                       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$:
011056  012737  071106  001174                MOV     #ZEROS,$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
```

```
3927                                    ;WAIT FOR THE FORMAT TO COMPLETE
3928  011160  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3929
3930                                    ;GO READ STATUS FOR FORMAT OPERATION
3931  011164  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3932  011170  000404                           BR      50$                     ;GO TO 50$ IF NO ERROR
3933  011172  000240                           NOP                     ;RETURN HERE IF ERROR
3934  011174  104000                           ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3935  011176  000137  011716                   JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3936  011202                                50$:
3937
3938                                    ;VERIFY NO ERRORS DURING FORMAT
3939  011202  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3940  011206  000405                           BR      60$                     ;GO TO 60$ IF NO ERROR
3941  011210  000240                           NOP                     ;RETURN HERE IF ERROR
3942  011212  104000                           ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3943  011214  004736                           JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3944  011216  000137  011716                   JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3945  011222                                60$:
3946
3947                                    ;MOVE LOOP ADDRESSES TO NEXT OPERATION
3948  011222  012737  011240  001122          MOV     #70$,$LPADR
3949  011230  012737  011240  001124          MOV     #70$,$LPERR
3950  011236  000410                           BR      80$                     ;SKIP TO WRITE OPERATION
3951
3952                                    ;;*****************************************************************
3953                                    ;LOOP #2        WRITE,WRITE CHECK DATA
3954
3955  011240                                70$:
3956
3957
3958                                    ;PREPARE DEVICE FOR WRITE OPERATION
3959  011240  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3960  011244  054130                           .WORD   054130  ;TASK DESCRIPTOR
3961  011246  000404                           BR      80$                     ;GO TO 80$ IF NO ERROR
3962  011250  000240                           NOP                     ;RETURN HERE IF ERROR
3963  011252  104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3964  011254  000137  011716                   JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3965  011260                                80$:
3966
3967
3968  011260                                120$:
3969
3970                                    ;WRITE DATA TO THE DRIVE
3971  011260  012737  107552  001404          MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
3972  011266  012737  177400  001402          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
3973  011274  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
3974  011302  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3975  011306  112722  000006                  MOVB    #RMDA,(R2)+
3976  011312  112722  000034                  MOVB    #RMDC,(R2)+
3977  011316  112722  000032                  MOVB    #RMOF,(R2)+
3978  011322  112722  000004                  MOVB    #RMBA,(R2)+
3979  011326  112722  000002                  MOVB    #RMWC,(R2)+
3980  011332  112722  000000                  MOVB    #RMCS1,(R2)+
3981  011336  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
3982
```

```
3983  011342  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3984  011346  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
3985  011350  000240                  NOP                     ;RETURN HERE IF ERROR
3986  011352  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3987  011354  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3988  011360                  130$:
3989
3990                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3991  011360  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3992
3993                          ;WAIT FOR WRITE COMMAND TO COMPLETE
3994  011364  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3995
3996                          ;GO READ STATUS FOR WRITE COMMAND
3997  011370  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3998  011374  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
3999  011376  000240                  NOP                     ;RETURN HERE IF ERROR
4000  011400  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4001  011402  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4002  011406                  140$:
4003
4004                          ;CHECK FOR ERRORS DURING WRITE OPERATION
4005  011406  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4006  011412  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
4007  011414  000240                  NOP                     ;RETURN HERE IF ERROR
4008  011416  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4009  011420  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4010  011422  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4011  011426                  150$:
4012  011426  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4013  011432  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
4014  011434  000240                  NOP                     ;RETURN HERE IF ERROR
4015  011436  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4016  011440  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4017  011442  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4018  011446                  160$:
4019
4020  011446                  170$:
4021  011446  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4022  011452  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
4023  011454  000240                  NOP                     ;RETURN HERE IF ERROR
4024  011456  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4025  011460  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4026  011462  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4027  011466                  180$:
4028
4029                          ;CHANGE LOOP ADDRESSES
4030  011466  012737  011504  001122  MOV     #190$,$LPADR
4031  011474  012737  011504  001124  MOV     #190$,$LPERR
4032  011502  000410                  BR      200$                    ;SKIP TO NEXT OPERATION
4033
4034                          ;;*****************************************************************
4035                          ;LOOP #3        WRITE CHECK DATA
4036
4037  011504                  190$:
4038
```

```
4039                                    ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
4040  011504  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4041  011510  054130                          .WORD   054130  ;TASK DESCRIPTOR
4042  011512  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
4043  011514  000240                          NOP                     ;RETURN HERE IF ERROR
4044  011516  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4045  011520  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4046  011524                          200$:
4047
4048  011524                          240$:
4049
4050                                    ;WRITE CHECK DATA DATA FROM DEVICE
4051  011524  012737  000051  001400          MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
4052  011532  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4053  011536  112722  000006                  MOVB    #RMDA,(R2)+
4054  011542  112722  000032                  MOVB    #RMOF,(R2)+
4055  011546  112722  000034                  MOVB    #RMDC,(R2)+
4056  011552  112722  000004                  MOVB    #RMBA,(R2)+
4057  011556  112722  000002                  MOVB    #RMWC,(R2)+
4058  011562  112722  000000                  MOVB    #RMCS1,(R2)+
4059  011566  112712  000200                  MOVB    #200,(R2)
4060
4061  011572  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4062  011576  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
4063  011600  000240                          NOP                     ;RETURN HERE IF ERROR
4064  011602  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4065  011604  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4066  011610                          250$:
4067
4068                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4069  011610  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4070
4071                                    ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
4072  011614  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4073
4074                                    ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
4075  011620  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4076  011624  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
4077  011626  000240                          NOP                     ;RETURN HERE IF ERROR
4078  011630  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4079  011632  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4080  011636                          260$:
4081
4082                                    ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
4083  011636  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4084  011642  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
4085  011644  000240                          NOP                     ;RETURN HERE IF ERROR
4086  011646  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4087  011650  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4088  011652  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4089  011656                          270$:
4090  011656  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4091  011662  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
4092  011664  000240                          NOP                     ;RETURN HERE IF ERROR
4093  011666  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4094  011670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4095  011672  000137  011716                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4096  011676                            280$:
4097
4098  011676                            290$:
4099  011676  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4100  011702  000405                            BR      300$                    ;GO TO 300$ IF NO ERROR
4101  011704  000240                            NOP                     ;RETURN HERE IF ERROR
4102  011706  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4103  011710  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4104  011712  000137  011716                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4105  011716                            300$:
4106
4107  011716                            350$:
4108                                    ;;*******************************************************************
4109                                    ;*TEST 6        WRITE, WRITE CHECK ZEROS W/ WCE ERROR
4110                                    ;;*******************************************************************
4111  011716                            TST6:
4112  011716  000004                            SCOPE                   ;SCOPE CALL
4113  011720  000240                            NOP                     ;START OF TEST
4114  011722  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4115  011726  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4116  011732  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4117  011736  012737  000006  001226            MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4118
4119                                    ;;*******************************************************************
4120                                    ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
4121
4122  011744                            10$:
4123
4124                                    ;PREPARE THE DEVICE FOR FORMAT OPERATION
4125  011744  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4126  011750  054130                            .WORD   054130  ;TASK DESCRIPTOR
4127  011752  000404                            BR      20$                     ;GO TO 20$ IF NO ERROR
4128  011754  000240                            NOP                     ;RETURN HERE IF ERROR
4129  011756  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4130  011760  000137  013134                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4131  011764                            20$:
4132
4133                                    ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4134  011764  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
4135  011772  012737  000000  001406            MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4136  012000  012737  107546  001404            MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4137  012006  012737  177376  001402            MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
4138  012014  012737  010000  001432            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4139  012022  012737  000063  001400            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4140
4141                                    ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4142
4143  012030  004737  040734                    JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4144  012034  000405                            BR      25$                     ;GO TO 25$ IF NO ERROR
4145  012036  104401  067464                    TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4146  012042  104000                            ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4147  012044  000137  013134                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4148  012050                            25$:
4149  012050  012737  071106  001174            MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
4150  012056  012737  000001  001176            MOV     #1,$TMP1                ;RANGE OF PATTERN
```

```
4151  012064  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4152
4153                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4154  012070  012702  001535                MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4155  012074  112722  000034                MOVB    #RMDC,(R2)+
4156  012100  112722  000006                MOVB    #RMDA,(R2)+
4157  012104  112722  000004                MOVB    #RMBA,(R2)+
4158  012110  112722  000002                MOVB    #RMWC,(R2)+
4159  012114  112722  000032                MOVB    #RMOF,(R2)+
4160  012120  112722  000000                MOVB    #RMCS1,(R2)+
4161  012124  112712  000200                MOVB    #200,(R2)               ;TERMINATE TABLE
4162  012130                          30$:
4163
4164                                  ;FORMAT THE DRIVE
4165  012130  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4166  012134  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
4167  012136  000240                        NOP                     ;RETURN HERE IF ERROR
4168  012140  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4169  012142  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4170  012146                          40$:
4171
4172                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4173  012146  004737  043452                JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
4174
4175                                  ;WAIT FOR THE FORMAT TO COMPLETE
4176  012152  004737  044346                JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
4177
4178                                  ;GO READ STATUS FOR FORMAT OPERATION
4179  012156  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4180  012162  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
4181  012164  000240                        NOP                     ;RETURN HERE IF ERROR
4182  012166  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4183  012170  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4184  012174                          50$:
4185
4186                                  ;VERIFY NO ERRORS DURING FORMAT
4187  012174  004737  057036                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4188  012200  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
4189  012202  000240                        NOP                     ;RETURN HERE IF ERROR
4190  012204  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4191  012206  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4192  012210  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4193  012214                          60$:
4194
4195                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
4196  012214  012737  012232  001122        MOV     #70$,$LPADR
4197  012222  012737  012232  001124        MOV     #70$,$LPERR
4198  012230  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
4199
4200                                  ;;*****************************************************************
4201                                  ;LOOP #2        WRITE,WRITE CHECK DATA
4202
4203  012232                          70$:
4204
4205
4206                                  ;PREPARE DEVICE FOR WRITE OPERATION
```

```
4207  012232  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4208  012236  054130                  .WORD   054130  ;TASK DESCRIPTOR
4209  012240  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
4210  012242  000240                  NOP                     ;RETURN HERE IF ERROR
4211  012244  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4212  012246  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4213  012252                  80$:
4214
4215
4216  012252                  120$:
4217
4218                          ;WRITE DATA TO THE DRIVE
4219  012252  012737  177400  001402          MOV     #<^C256.*1>,RMWCO       ;CHANGE WORD COUNT
4220  012260  012737  107552  001404          MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
4221  012266  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
4222  012274  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4223  012300  112722  000006                  MOVB    #RMDA,(R2)+
4224  012304  112722  000034                  MOVB    #RMDC,(R2)+
4225  012310  112722  000032                  MOVB    #RMOF,(R2)+
4226  012314  112722  000004                  MOVB    #RMBA,(R2)+
4227  012320  112722  000002                  MOVB    #RMWC,(R2)+
4228  012324  112722  000000                  MOVB    #RMCS1,(R2)+
4229  012330  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
4230
4231  012334  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4232  012340  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
4233  012342  000240                  NOP                     ;RETURN HERE IF ERROR
4234  012344  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4235  012346  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4236  012352                  130$:
4237
4238                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4239  012352  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4240
4241                          ;WAIT FOR WRITE COMMAND TO COMPLETE
4242  012356  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4243
4244                          ;GO READ STATUS FOR WRITE COMMAND
4245  012362  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4246  012366  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
4247  012370  000240                  NOP                     ;RETURN HERE IF ERROR
4248  012372  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4249  012374  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4250  012400                  140$:
4251
4252                          ;CHECK FOR ERRORS DURING WRITE OPERATION
4253  012400  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4254  012404  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
4255  012406  000240                  NOP                     ;RETURN HERE IF ERROR
4256  012410  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4257  012412  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4258  012414  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4259  012420                  150$:
4260  012420  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4261  012424  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
4262  012426  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
                                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     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR

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

                                190$:

                                        MOV     #1,R3                   ;R3=WCE BIT POSITION
                                195$:   BIS     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER

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

                                240$:

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

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      250$                    ;GO TO 250$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                250$:
```

```
4319
4320  012616  004737  043452          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4321
4322
4323  012622  004737  044346          ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4324
4325
4326  012626  004737  043536          ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4327  012632  000404                  BR      260$                    ;GO TO 260$ IF NO ERROR
4328  012634  000240                  NOP                     ;RETURN HERE IF ERROR
4329  012636  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4330  012640  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4331  012644                  260$:
4332
4333                          ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
4334  012644  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4335  012650  000405                  BR      270$                    ;GO TO 270$ IF NO ERROR
4336  012652  000240                  NOP                     ;RETURN HERE IF ERROR
4337  012654  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4338  012656  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4339  012660  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4340  012664                  270$:
4341  012664  032737  040000  001340      BIT     #WCE,RMCS2I             ;WAS "WCE" DETECTED??
4342  012672  001030                  BNE     285$                    ;YES!!
4343  012674  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4344  012700  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
4345  012702  000240                  NOP                     ;RETURN HERE IF ERROR
4346  012704  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4347  012706  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4348  012710  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4349  012714                  280$:
4350  012714  013737  001340  001140      MOV     RMCS2I,$GDDAT           ;EXPECTED STATUS
4351  012722  052737  040000  001140      BIS     #WCE,$GDDAT
4352  012730  013737  001340  001142      MOV     RMCS2I,$BDDAT           ;RECEIVED STATUS
4353  012736  010337  001174              MOV     R3,$TMP0                ;FAILING BIT POSITION
4354  012742  012737  110550  001176      MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
4355  012750  104337                      ERROR   337                     ;WCE NOT DETECTED
4356  012752  000470                      BR      350$
4357
4358  012754                  285$:
4359  012754  112737  000022  001506      MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
4360  012762  112737  000200  001507      MOVB    #200,GETINX+1
4361  012770  004737  043536              JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
4362  012774  000404                      BR      290$            ;GO TO 290$ IF NO ERROR
4363  012776  000240                      NOP                     ;RETURN HERE IF ERROR
4364  013000  104000                      ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
4365  013002  000137  013134              JMP     350$            ;GO TO 350$ IF ERROR
4366
4367  013006                  290$:
4368  013006  013737  001352  001142      MOV     RMDBI,$BDDAT            ;RECEIVED DATA
4369  013014  013737  110550  001140      MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
4370  013022  040337  001140              BIC     R3,$GDDAT
4371  013026  012737  110550  001134      MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
4372  013034  013737  001334  001136      MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
4373  013042  162737  000002  001136      SUB     #2,$BDADR               ;ADJUST BUS ADDRESS
4374  013050  023737  001134  001136      CMP     $GDADR,$BDADR           ;ADDRESSES OK??
```

```
4375  013056  001402                          BEQ     295$                    ;YES!!
4376  013060  104340                          ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
4377  013062  000424                          BR      350$
4378  013064  023737  001140  001142  295$:   CMP     $GDDAT,$BDDAT           ;DATA OK??
4379  013072  001402                          BEQ     296$                    ;YES!!
4380  013074  104341                          ERROR   341                     ;UNEXPECTED WCE DATA
4381  013076  000416                          BR      350$
4382  013100                          296$:
4383
4384  013100  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4385  013104  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
4386  013106  000240                          NOP                     ;RETURN HERE IF ERROR
4387  013110  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4388  013112  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4389  013114  000137  013134                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4390  013120                          300$:
4391
4392  013120  040337  110550                  BIC     R3,BUFTWO-2             ;RESTORE DATA PATTERN
4393  013124  006303                          ASL     R3                      ;SHIFT TO NEXT BIT
4394  013126  001402                          BEQ     350$                    ;EXIT IF DONE
4395  013130  000137  012506                  JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
4396  013134                          350$:
4397  013134  012737  000000  001210          MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
4398  013142  012737  011744  001122          MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
4399  013150  012737  011744  001124          MOV     #10$,$LPERR
4400                          ;;*****************************************************************
4401                          ;*TEST 7         WRITE, READ ONES
4402                          ;;*****************************************************************
4403  013156                  TST7:
4404  013156  000004                          SCOPE                   ;SCOPE CALL
4405  013160  000240                          NOP                     ;START OF TEST
4406  013162  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4407  013166  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4408  013172  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4409  013176  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4410
4411                          ;;*****************************************************************
4412                          ;LOOP #1        FORMAT,WRITE,READ
4413
4414  013204                  10$:
4415
4416                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
4417  013204  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4418  013210  054130                  .WORD   054130  ;TASK DESCRIPTOR
4419  013212  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
4420  013214  000240                  NOP                     ;RETURN HERE IF ERROR
4421  013216  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4422  013220  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4423  013224                  20$:
4424
4425                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4426  013224  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4427  013232  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4428  013240  012737  107546  001404          MOV     #BUF0NE,RMBA0           ;BUS ADDRESS
4429  013246  012737  177400  001402          MOV     #<^C256.*1>,RMWC0       ;WORD COUNT = 1 SECTOR
4430  013254  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
```

```
4431  013262  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4432
4433                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4434
4435  013270  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4436  013274  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
4437  013276  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4438  013302  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4439  013304  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4440  013310                          25$:
4441  013310  012737  071044  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4442  013316  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
4443  013324  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4444
4445                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
4446  013330  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4447  013334  112722  000034                  MOVB    #RMDC,(R2)+
4448  013340  112722  000006                  MOVB    #RMDA,(R2)+
4449  013344  112722  000004                  MOVB    #RMBA,(R2)+
4450  013350  112722  000002                  MOVB    #RMWC,(R2)+
4451  013354  112722  000032                  MOVB    #RMOF,(R2)+
4452  013360  112722  000000                  MOVB    #RMCS1,(R2)+
4453  013364  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
4454  013370                          30$:
4455
4456                                  ,FORMAT THE DRIVE
4457  013370  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4458  013374  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
4459  013376  000240                          NOP                     ;RETURN HERE IF ERROR
4460  013400  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4461  013402  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4462  013406                          40$:
4463
4464                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4465  013406  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4466
4467                                  ;WAIT FOR THE FORMAT TO COMPLETE
4468  013412  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4469
4470                                  ;GO READ STATUS FOR FORMAT OPERATION
4471  013416  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4472  013422  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
4473  013424  000240                          NOP                     ;RETURN HERE IF ERROR
4474  013426  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4475  013430  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4476  013434                          50$:
4477
4478                                  ;VERIFY NO ERRORS DURING FORMAT
4479  013434  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4480  013440  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
4481  013442  000240                          NOP                     ;RETURN HERE IF ERROR
4482  013444  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4483  013446  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4484  013450  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4485  013454                          60$:
4486
```

```
                                        ;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  107552  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$:
```

```
4543
4544                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
4545  013640  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4546  013644  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
4547  013646  000240                            NOP                     ;RETURN HERE IF ERROR
4548  013650  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4549  013652  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4550  013654  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4551  013660                                150$:
4552  013660  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4553  013664  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
4554  013666  000240                            NOP                     ;RETURN HERE IF ERROR
4555  013670  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4556  013672  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4557  013674  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4558  013700                                160$:
4559
4560  013700                                170$:
4561  013700  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4562  013704  000405                            BR      180$                    ;GO TO 180$ IF NO ERROR
4563  013706  000240                            NOP                     ;RETURN HERE IF ERROR
4564  013710  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBRO' INE
4565  013712  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4566  013714  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS ,OUND
4567  013720                                180$:
4568
4569                                    ;CHANGE LOOP ADDRESSES
4570  013720  012737  013736  001122            MOV     #190$,$LPADR
4571  013726  012737  013736  001124            MOV     #190$,$LPERR
4572  013734  000410                            BR      200$                    ;SKIP TO NEXT OPERATION
4573
4574                                    ;;*****************************************************************
4575                                    ;LOOP #3        READ
4576
4577  013736                                190$:
4578
4579                                    ;PREPARE DEVICE FOR READ OPERATION
4580  013736  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4581  013742  054130                            .WORD   054130  ;TASK DESCRIPTOR
4582  013744  000404                            BR      200$                    ;GO TO 200$ IF NO ERROR
4583  013746  000240                            NOP                     ;RETURN HERE IF ERROR
4584  013750  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4585  013752  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4586  013756                                200$:
4587
4588  013756                                240$:
4589
4590                                    ;READ DATA FROM DEVICE
4591  013756  012737  110556  001404            MOV     #BUFTWO+4,RMBAO         ;CHANGE MEMORY ADDRESS
4592  013764  012737  000071  001400            MOV     #RD!GO,RMCS1O           ;READ DATA COMMAND
4593  013772  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4594  013776  112722  000006                    MOVB    #RMDA,(R2)+
4595  014002  112722  000032                    MOVB    #RMOF,(R2)+
4596  014006  112722  000034                    MOVB    #RMDC,(R2)+
4597  014012  112722  000004                    MOVB    #RMBA,(R2)+
4598  014016  112722  000002                    MOVB    #RMWC,(R2)+
```

```
4599  014022  112722  000000          MOVB    #RMCS1,(R2)+
4600  014026  112712  000200          MOVB    #200,(R2)
4601
4602  014032  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4603  014036  000404                  BR      250$                    ;GO TO 250$ IF NO ERROR
4604  014040  000240                  NOP                     ;RETURN HERE IF ERROR
4605  014042  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4606  014044  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4607  014050                  250$:
4608
4609                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4610  014050  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4611
4612                          ;WAIT FOR READ OPERATION TO COMPLETE
4613  014054  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4614
4615                          ;GO READ STATUS FOR READ OPERATION
4616  014060  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4617  014064  000404                  BR      260$                    ;GO TO 260$ IF NO ERROR
4618  014066  000240                  NOP                     ;RETURN HERE IF ERROR
4619  014070  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4620  014072  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4621  014076                  260$:
4622
4623                          ;CHECK FOR ERRORS DURING READ OPERATION
4624  014076  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4625  014102  000405                  BR      270$                    ;GO TO 270$ IF NO ERROR
4626  014104  000240                  NOP                     ;RETURN HERE IF ERROR
4627  014106  104000                  ERROP                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4628  014110  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4629  014112  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4630  014116                  270$:
4631  014116  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4632  014122  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
4633  014124  000240                  NOP                     ;RETURN HERE IF ERROR
4634  014126  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4635  014130  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4636  014132  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4637  014136                  280$:
4638
4639  014136                  290$:
4640  014136  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4641  014142  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
4642  014144  000240                  NOP                     ;RETURN HERE IF ERROR
4643  014146  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4644  014150  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4645  014152  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4646  014156                  300$:
4647  014156  004737  043104          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
4648  014162  107552                  .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
4649  014164  110556                  .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
4650  014166  000404                  BR      310$                    ;GO TO 310$ IF NO ERROR
4651  014170  000240                  NOP                     ;RETURN HERE IF ERROR
4652  014172  104000                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4653  014174  000137  014200          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4654  014200                  310$:
```

```
4655
4656  014200                          350$:
4657                                  ;;*****************************************************************
4658                                  ;*TEST 10       WRITE, WRITE CHECK ONES
4659                                  ;;*****************************************************************
4660  014200                          TST10:
4661  014200  000004                         SCOPE                   ;SCOPE CALL
4662  014202  000240                         NOP                     ;START OF TEST
4663  014204  012706  001100                 MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4664  014210  013700  001276                 MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4665  014214  013701  001450                 MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4666  014220  012737  000010  001226         MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4667
4668                                  ;;*****************************************************************
4669                                  ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
4670
4671  014226                          10$:
4672
4673                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
4674  014226  004737  040020                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4675  014232  054130                         .WORD   054130  ;TASK DESCRIPTOR
4676  014234  000404                         BR      20$                     ;GO TO 20$ IF NO ERROR
4677  014236  000240                         NOP                     ;RETURN HERE IF ERROR
4678  014240  104000                         ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4679  014242  000137  015172                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4680  014246                          20$:
4681
4682                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4683  014246  012737  000000  001434         MOV     #0,RMDC0                ;CYLINDER = 0
4684  014254  012737  000000  001406         MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4685  014262  012737  107546  001404         MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4686  014270  012737  177376  001402         MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
4687  014276  012737  010000  001432         MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4688  014304  012737  000063  001400         MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4689
4690                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4691
4692  014312  004737  040734                 JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4693  014316  000405                         BR      25$                     ;GO TO 25$ IF NO ERROR
4694  014320  104401  067464                 TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4695  014324  104000                         ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4696  014326  000137  015172                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4697  014332                          25$:
4698  014332  012737  071044  001174         MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4699  014340  012737  000001  001176         MOV     #1,$TMP1                ;RANGE OF PATTERN
4700  014346  004737  042640                 JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4701
4702                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4703  014352  012702  001535                 MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4704  014356  112722  000034                 MOVB    #RMDC,(R2)+
4705  014362  112722  000006                 MOVB    #RMDA,(R2)+
4706  014366  112722  000004                 MOVB    #RMBA,(R2)+
4707  014372  112722  000002                 MOVB    #RMWC,(R2)+
4708  014376  112722  000032                 MOVB    #RMOF,(R2)+
4709  014402  112722  000000                 MOVB    #RMCS1,(R2)+
4710  014406  112712  000200                 MOVB    #200,(R2)               ;TERMINATE TABLE
```

```
4711  014412                          30$:
4712
4713                                  ;FORMAT THE DRIVE
4714  014412  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4715  014416  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
4716  014420  000240                        NOP                     ;RETURN HERE IF ERROR
4717  014422  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4718  014424  000137  015172                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4719  014430                          40$:
4720
4721                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4722  014430  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4723
4724                                  ;WAIT FOR THE FORMAT TO COMPLETE
4725  014434  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4726
4727                                  ;GO READ STATUS FOR FORMAT OPERATION
4728  014440  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4729  014444  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
4730  014446  000240                        NOP                     ;RETURN HERE IF ERROR
4731  014450  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4732  014452  000137  015172                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4733  014456                          50$:
4734
4735                                  ;VERIFY NO ERRORS DURING FORMAT
4736  014456  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4737  014462  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
4738  014464  000240                        NOP                     ;RETURN HERE IF ERROR
4739  014466  104000                        ERROP                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4740  014470  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4741  014472  000137  015172                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4742  014476                          60$:
4743
4744                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
4745  014476  012737  014514  001122        MOV     #70$,$LPADR
4746  014504  012737  014514  001124        MOV     #70$,$LPERR
4747  014512  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
4748
4749                                  ;;*****************************************************************
4750                                  ;LOOP #2        WRITE,WRITE CHECK DATA
4751
4752  014514                          70$:
4753
4754
4755                                  ;PREPARE DEVICE FOR WRITE OPERATION
4756  014514  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4757  014520  054130                        .WORD   054130  ;TASK DESCRIPTOR
4758  014522  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
4759  014524  000240                        NOP                     ;RETURN HERE IF ERROR
4760  014526  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4761  014530  000137  015172                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4762  014534                          80$:
4763
4764
4765  014534                          120$:
4766
```

```
                                        ;WRITE DATA TO THE DRIVE
014534  012737  107552  001404            MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
014542  012737  177400  001402            MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
014550  012737  000061  001400            MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
014556  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
014562  112722  000006                    MOVB    #RMDA,(R2)+
014566  112722  000034                    MOVB    #RMDC,(R2)+
014572  112722  000032                    MOVB    #RMOF,(R2)+
014576  112722  000004                    MOVB    #RMBA,(R2)+
014602  112722  000002                    MOVB    #RMWC,(R2)+
014606  112722  000000                    MOVB    #RMCS1,(R2)+
014612  112722  000200                    MOVB    #200,(R2)+              ;TERMINATE TABLE

014616  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
014622  000404                            BR      130$                    ;GO TO 130$ IF NO ERROR
014624  000240                            NOP                     ;RETURN HERE IF ERROR
014626  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
014630  000137  015172                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014634                                  130$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
014634  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
014640  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR WRITE COMMAND
014644  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
014650  000404                            BR      140$                    ;GO TO 140$ IF NO ERROR
014652  000240                            NOP                     ;RETURN HERE IF ERROR
014654  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
014656  000137  015172                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014662                                  140$:

                                        ;CHECK FOR ERRORS DURING WRITE OPERATION
014662  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
014666  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
014670  000240                            NOP                     ;RETURN HERE IF ERROR
014672  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
014674  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
014676  000137  015172                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014702                                  150$:
014702  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
014706  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
014710  000240                            NOP                     ;RETURN HERE IF ERROR
014712  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
014714  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
014716  000137  015172                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014722                                  160$:

014722                                  170$:
014722  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
014726  000405                            BR      180$                    ;GO TO 180$ IF NO ERROR
014730  000240                            NOP                     ;RETURN HERE IF ERROR
014732  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
014734  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4823  014736  000137  015172          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4824  014742                        180$:
4825
4826                                ;CHANGE LOOP ADDRESSES
4827  014742  012737  014760  001122        MOV     #190$,$LPADR
4828  014750  012737  014760  001124        MOV     #190$,$LPERR
4829  014756  000410                        BR      200$                    ;SKIP TO NEXT OPERATION
4830
4831                                ;;*****************************************************************
4832                                ;LOOP #3        WRITE CHECK DATA
4833
4834  014760                        190$:
4835
4836                                ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
4837  014760  004737  040020               JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4838  014764  054130                        .WORD   054130  ;TASK DESCRIPTOR
4839  014766  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
4840  014770  000240                        NOP                     ;RETURN HERE IF ERROR
4841  014772  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4842  014774  000137  015172          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4843  015000                        200$:
4844
4845  015000                        240$:
4846
4847                                ;WRITE CHECK DATA DATA FROM DEVICE
4848  015000  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
4849  015006  012702  001535              MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4850  015012  112722  000006              MOVB    #RMDA,(R2)+
4851  015016  112722  000032              MOVB    #RMOF,(R2)+
4852  015022  112722  000034              MOVB    #RMDC,(R2)+
4853  015026  112722  000004              MOVB    #RMBA,(P2)+
4854  015032  112722  000002              MOVB    #RMWC,(R2)+
4855  015036  112722  000000              MOVB    #RMCS1,(R2)+
4856  015042  112712  000200              MOVB    #200,(R2)
4857
4858  015046  004737  044006              JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4859  015052  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
4860  015054  000240                        NOP                     ;RETURN HERE IF ERROR
4861  015056  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4862  015060  000137  015172          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4863  015064                        250$:
4864
4865                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4866  015064  004737  043452               JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4867
4868                                ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
4869  015070  004737  044346               JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4870
4871                                ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
4872  015074  004737  043536               JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4873  015100  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
4874  015102  000240                        NOP                     ;RETURN HERE IF ERROR
4875  015104  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4876  015106  000137  015172          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4877  015112                        260$:
4878
```

```
                                ;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$:

015152                          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$:

015172                          350$:
                                ;;*****************************************************************
                                ;*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  000137  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  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
015262  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
```

```
4935  015270  012737  010000  001432          MOV     #FMT16,RMOFO            ;16 BIT FORMAT
4936  015276  012737  000063  001400          MOV     #WH!GO,RMCS1O           ;WRITE HEADER AND DATA COMMAND
4937
4938                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4939
4940  015304  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4941  015310  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
4942  015312  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4943  015316  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4944  015320  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4945  015324                          25$:
4946  015324  012737  071044  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4947  015332  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
4948  015340  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4949
4950                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4951  015344  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4952  015350  112722  000034                  MOVB    #RMDC,(R2)+
4953  015354  112722  000006                  MOVB    #RMDA,(R2)+
4954  015360  112722  000004                  MOVB    #RMBA,(R2)+
4955  015364  112722  000002                  MOVB    #RMWC,(R2)+
4956  015370  112722  000032                  MOVB    #RMOF,(R2)+
4957  015374  112722  000000                  MOVB    #RMCS1,(R2)+
4958  015400  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
4959  015404                          30$:
4960
4961                                  ;FORMAT THE DRIVE
4962  015404  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4963  015410  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
4964  015412  000240                          NOP                     ;RETURN HERE IF ERROR
4965  015414  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4966  015416  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4967  015422                          40$:
4968
4969                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4970  015422  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4971
4972                                  ;WAIT FOR THE FORMAT TO COMPLETE
4973  015426  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4974
4975                                  ;GO READ STATUS FOR FORMAT OPERATION
4976  015432  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4977  015436  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
4978  015440  000240                          NOP                     ;RETURN HERE IF ERROR
4979  015442  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4980  015444  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4981  015450                          50$:
4982
4983                                  ;VERIFY NO ERRORS DURING FORMAT
4984  015450  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4985  015454  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
4986  015456  000240                          NOP                     ;RETURN HERE IF ERROR
4987  015460  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4988  015462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4989  015464  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4990  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  107552  001404            MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
015542  012737  000061  001400            MOV     #WD!GO,RMCS1O           ;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
```

```
5047  015654                          140$:
5048
5049                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
5050  015654  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5051  015660  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5052  015662  000240                        NOP                     ;RETURN HERE IF ERROR
5053  015664  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5054  015666  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5055  015670  000137  016412                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5056  015674                          150$:
5057  015674  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5058  015700  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5059  015702  000240                        NOP                     ;RETURN HERE IF ERROR
5060  015704  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5061  015706  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5062  015710  000137  016412                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5063  015714                          160$:
5064
5065  015714                          170$:
5066  015714  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5067  015720  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
5068  015722  000240                        NOP                     ;RETURN HERE IF ERROR
5069  015724  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5070  015726  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5071  015730  000137  016412                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5072  015734                          180$:
5073
5074                                  ;CHANGE LOOP ADDRESSES
5075  015734  012737  015756  001122        MOV     #190$,$LPADR
5076  015742  012737  015756  001124        MOV     #190$,$LPERR
5077  015750  012737  015762  001210        MOV     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR
5078
5079                                  ;;**********************************************************************
5080                                  ;LOOP #3        WRITE CHECK DATA
5081
5082  015756                          190$:
5083
5084  015756  012703  000001                MOV     #1,R3                   ;R3+WCE BIT POSITION
5085  015762  040337  110550          195$: BIC     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER
5086
5087                                  ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
5088  015766  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5089  015772  054130                        .WORD   054130  ;TASK DESCRIPTOR
5090  015774  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5091  015776  000240                        NOP                     ;RETURN HERE IF ERROR
5092  016000  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5093  016002  000137  016412                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5094  016006                          200$:
5095
5096  016006                          240$:
5097
5098                                  ;READ DATA FROM DEVICE
5099  016006  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
5100  016014  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5101  016020  112722  000006                MOVB    #RMDA,(R2)+
5102  016024  112722  000032                MOVB    #RMOF,(R2)+
```

```
5103  016030  112722  000034          MOVB    #RMDC,(R2)+
5104  016034  112722  000004          MOVB    #RMBA,(R2)+
5105  016040  112722  000002          MOVB    #RMWC,(R2)+
5106  016044  112722  000000          MOVB    #RMCS1,(R2)+
5107  016050  112712  000200          MOVB    #200,(R2)
5108
5109  016054  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5110  016060  000404                  BR      250$                    ;GO TO 250$ IF NO ERROR
5111  016062  000240                  NOP                     ;RETURN HERE IF ERROR
5112  016064  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5113  016066  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5114  016072                  250$:
5115
5116                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5117  016072  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5118
5119                          ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
5120  016076  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5121
5122                          ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
5123  016102  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5124  016106  000404                  BR      260$                    ;GO TO 260$ IF NO ERROR
5125  016110  000240                  NOP                     ;RETURN HERE IF ERROR
5126  016112  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5127  016114  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5128  016120                  260$:
5129
5130                          ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
5131  016120  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5132  016124  000405                  BR      270$                    ;GO TO 270$ IF NO ERROR
5133  016126  000240                  NOP                     ;RETURN HERE IF ERROR
5134  016130  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5135  016132  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5136  016134  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5137  016140                  270$:
5138  016140  032737  040000  001340          BIT     #WCE,RMCS21             ;WAS "WCE" DETECTED??
5139  016146  001030                  BNE     285$                    ;YES!!
5140  016150  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5141  016154  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
5142  016156  000240                  NOP                     ;RETURN HERE IF ERROR
5143  016160  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5144  016162  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5145  016164  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5146  016170                  280$:
5147  016170  013737  001340  001140          MOV     RMCS21,$GDDAT           ;EXPECTED STATUS
5148  016176  052737  040000  001140          BIS     #WCE,$GDDAT
5149  016204  013737  001340  001142          MOV     RMCS21,$BDDAT           ;RECEIVED STATUS
5150  016212  010337  001174                  MOV     R3,$TMP0                ;FAILING BIT POSITION
5151  016216  012737  110550  001176          MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
5152  016224  104337                  ERROR   337                     ;WCE NOT DETECTED
5153  016226  000471                  BR      350$
5154
5155  016230                  285$:
5156  016230  112737  000022  001506          MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
5157  016236  112737  000200  001507          MOVB    #200,GETINX+1
5158  016244  012737  016412  001352          MOV     #350$,RMDB1     ;SET THE INPUT BUFFER
```

```
5159  016252  004737  043536          JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
5160  016256  000402                  BR      290$            ;GO TO 290$ IF NO ERROR
5161  016260  000240                  NOP                     ;RETURN HERE IF ERROR
5162  016262  104000                  ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
5163
5164  016264                  290$:
5165  016264  013737  001352  001142  MOV     RMDBI,$BDDAT            ;RECEIVED DATA
5166  016272  013737  110550  001140  MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
5167  016300  050337  001140          BIS     R3,$GDDAT
5168  016304  012737  110550  001134  MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
5169  016312  013737  001334  001136  MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
5170  016320  162737  000002  001136  SUB     #2,$BDADR               ;CORRECT MEMORY ADDRESS
5171  016326  023737  001134  001136  CMP     $GDADR,$BDADR           ;ADDRESSES OK??
5172  016334  001402                  BEQ     295$                    ;YES!!
5173  016336  104340                  ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
5174  016340  000424                  BR      350$
5175  016342  023737  001140  001142  295$:   CMP     $GDDAT,$BDDAT           ;DATA OK??
5176  016350  001402                  BEQ     296$                    ;YES!!
5177  016352  104341                  ERROR   341                     ;UNEXPECTED WCE DATA
5178  016354  000416                  BR      350$
5179  016356                  296$:
5180
5181  016356  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5182  016362  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
5183  016364  000240                  NOP                     ;RETURN HERE IF ERROR
5184  016366  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5185  016370  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5186  016372  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5187  016376                  300$:
5188
5189  016376  050337  110550          BIS     R3,BUFTWO-2             ;RESTORE DATA PATTERN
5190  016402  006303                  ASL     R3                      ;SHIFT TO NEXT BIT
5191  016404  001402                  BEQ     350$                    ;EXIT IF DONE
5192  016406  000137  015762          JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
5193  016412                  350$:
5194  016412  012737  000000  001210  MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5195  016420  012737  015220  001122  MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
5196  016426  012737  015220  001124  MOV     #10$,$LPERR
5197                          ;;*****************************************************************
5198                          ;*TEST 12        WRITE, WRITE CHECK MULTIPLE SECTORS
5199                          ;;*****************************************************************
5200  016434                  TST12:
5201  016434  000004                  SCOPE                   ;SCOPE CALL
5202  016436  000240                  NOP                     ;START OF TEST
5203  016440  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5204  016444  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5205  016450  013701  001450          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5206  016454  012737  000012  001226  MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5207
5208                          ;;*****************************************************************
5209                          ;LOOP #1        FORMAT,WRITE,READ
5210
5211  016462                  10$:
5212
5213                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
5214  016462  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
5215  016466  054130                  .WORD   054130  ;TASK DESCRIPTOR
5216  016470  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
5217  016472  000240                  NOP                     ;RETURN HERE IF ERROR
5218  016474  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5219  016476  000137  017454          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5220  016502                  20$:
5221
5222                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5223  016502  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
5224  016510  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
5225  016516  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5226  016524  012737  177240  001402          MOV     #<^C<2**<2+256>>+1>,RMWC0        ;WORD COUNT
5227  016532  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5228  016540  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5229
5230                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5231
5232  016546  004737  040734          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5233  016552  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
5234  016554  104401  067464          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5235  016560  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5236  016562  000137  017454          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5237  016566                  25$:
5238  016566  012737  071106  001174          MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
5239  016574  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
5240  016602  004737  042640          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5241
5242                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5243  016606  012702  001535          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5244  016612  112722  000034          MOVB    #RMDC,(R2)+
5245  016616  112722  000006          MOVB    #RMDA,(R2)+
5246  016622  112722  000004          MOVB    #RMBA,(R2)+
5247  016626  112722  000002          MOVB    #RMWC,(R2)+
5248  016632  112722  000032          MOVB    #RMOF,(R2)+
5249  016636  112722  000000          MOVB    #RMCS1,(R2)+
5250  016642  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
5251  016646                  30$:
5252
5253                          ;FORMAT THE DRIVE
5254  016646  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5255  016652  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
5256  016654  000240                  NOP                     ;RETURN HERE IF ERROR
5257  016656  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5258  016660  000137  017454          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5259  016664                  40$:
5260
5261                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
5262  016664  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5263
5264                          ;WAIT FOR THE FORMAT TO COMPLETE
5265  016670  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5266
5267                          ;GO READ STATUS FOR FORMAT OPERATION
5268  016674  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5269  016700  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
5270  016702  000240                  NOP                     ;RETURN HERE IF ERROR
```

```
5271  016704  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5272  016706  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5273  016712                          50$:
5274
5275                                  ;VERIFY NO ERRORS DURING FORMAT
5276  016712  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5277  016716  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
5278  016720  000240                          NOP                     ;RETURN HERE IF ERROR
5279  016722  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5280  016724  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5281  016726  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5282  016732                          60$:
5283
5284                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5285  016732  012737  017012  001122          MOV     #70$,$LPADR
5286  016740  012737  017012  001124          MOV     #70$,$LPERR
5287                                  ;REGENERATE DATA BUFFER
5288  016746  012737  177000  001402          MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
5289  016754  012737  107546  001404          MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRESS
5290  016762  012737  000060  001400          MOV     #WD,RMCS1O              ;WRITE DATA
5291  016770  012737  071044  001174          MOV     #ONES,$TMPO             ;STARTING ADDRESS OF PATTERN
5292  016776  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
5293  017004  004737  042640                  JSR     PC,GENBUF
5294  017010  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
5295
5296                                  ;;*****************************************************************
5297                                  ;LOOP #2        WRITE,READ
5298
5299  017012                          70$:
5300
5301
5302                                  ;PREPARE DEVICE FOR WRITE OPERATION
5303  017012  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5304  017016  054130                          .WORD   054130  ;TASK DESCRIPTOR
5305  017020  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
5306  017022  000240                          NOP                     ;RETURN HERE IF ERROR
5307  017024  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5308  017026  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5309  017032                          80$:
5310
5311
5312  017032                          120$:
5313
5314                                  ;WRITE DATA TO THE DRIVE
5315  017032  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
5316  017040  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5317  017044  112722  000006                  MOVB    #RMDA,(R2)+
5318  017050  112722  000034                  MOVB    #RMDC,(R2)+
5319  017054  112722  000032                  MOVB    #RMOF,(R2)+
5320  017060  112722  000004                  MOVB    #RMBA,(R2)+
5321  017064  112722  000002                  MOVB    #RMWC,(R2)+
5322  017070  112722  000000                  MOVB    #RMCS1,(R2)+
5323  017074  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
5324
5325  017100  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5326  017104  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
```

```
5327  017106  000240                          NOP                     ;RETURN HERE IF ERROR
5328  017110  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5329  017112  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5330  017116                          130$:
5331
5332                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5333  017116  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5334
5335                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
5336  017122  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5337
5338                                  ;GO READ STATUS FOR WRITE COMMAND
5339  017126  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5340  017132  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
5341  017134  000240                          NOP                     ;RETURN HERE IF ERROR
5342  017136  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5343  017140  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5344  017144                          140$:
5345
5346                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
5347  017144  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5348  017150  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5349  017152  000240                          NOP                     ;RETURN HERE IF ERROR
5350  017154  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5351  017156  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5352  017160  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5353  017164                          150$:
5354  017164  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5355  017170  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5356  017172  000240                          NOP                     ;RETURN HERE IF ERROR
5357  017174  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5358  017176  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5359  017200  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5360  017204                          160$:
   1
   2  017204                          170$:
5363  017204  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5364  017210  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5365  017212  000240                          NOP                     ;RETURN HERE IF ERROR
5366  017214  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5367  017216  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5368  017220  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5369  017224                          180$:
5370
5371                                  ;CHANGE LOOP ADDRESSES
5372  017224  012737  017242  001122            MOV     #190$,$LPADR
5373  017232  012737  017242  001124            MOV     #190$,$LPERR
5374  017240  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
5375
5376                                  ;;*****************************************************************
5377                                  ;LOOP #3        READ
5378
5379  017242                          190$:
5380
5381                                  ;PREPARE DEVICE FOR READ OPERATION
5382  017242  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
5383  017246  054130                          .WORD   054130  ;TASK DESCRIPTOR
5384  017250  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
5385  017252  000240                          NOP                     ;RETURN HERE IF ERROR
5386  017254  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5387  017256  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5388  017262                          200$:
5389
5390  017262                          240$:
5391
5392                                  ;READ DATA FROM DEVICE
5393  017262  012737  000051  001400          MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5394  017270  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5395  017274  112722  000006                  MOVB    #RMDA,(R2)+
5396  017300  112722  000032                  MOVB    #RMOF,(R2)+
5397  017304  112722  000034                  MOVB    #RMDC,(R2)+
5398  017310  112722  000004                  MOVB    #RMBA,(R2)+
5399  017314  112722  000002                  MOVB    #RMWC,(R2)+
5400  017320  112722  000000                  MOVB    #RMCS1,(R2)+
5401  017324  112712  000200                  MOVB    #200,(R2)
5402
5403  017330  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5404  017334  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
5405  017336  000240                          NOP                     ;RETURN HERE IF ERROR
5406  017340  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5407  017342  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5408  017346                          250$:
5409
5410                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5411  017346  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5412
5413                                  ;WAIT FOR READ OPERATION TO COMPLETE
5414  017352  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5415
5416                                  ;GO READ STATUS FOR READ OPERATION
5417  017356  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5418  017362  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
5419  017364  000240                          NOP                     ;RETURN HERE IF ERROR
5420  017366  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5421  017370  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5422  017374                          260$:
5423
5424                                  ;CHECK FOR ERRORS DURING READ OPERATION
5425  017374  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5426  017400  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
5427  017402  000240                          NOP                     ;RETURN HERE IF ERROR
5428  017404  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5429  017406  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5430  017410  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5431  017414                          270$:
5432  017414  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5433  017420  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
5434  017422  000240                          NOP                     ;RETURN HERE IF ERROR
5435  017424  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5436  017426  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5437  017430  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5438  017434                          280$:
```

```
5439
5440  017434                                290$:
5441  017434  004737  045364                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5442  017440  000405                                BR      300$                    ;GO TO 300$ IF NO ERROR
5443  017442  000240                                NOP                     ;RETURN HERE IF ERROR
5444  017444  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5445  017446  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5446  017450  000137  017454                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5447  017454                                300$:
5448
5449  017454                                350$:
5450                                        ;;*****************************************************************
5451                                        ;*TEST 13       WRITE, READ W/ IMPLIED SEEK
5452                                        ;;*****************************************************************
5453  017454                                TST13:
5454  017454  000004                                SCOPE                   ;SCOPE CALL
5455  017456  000240                                NOP                     ;START OF TEST
5456  017460  012706  001100                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5457  017464  013700  001276                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5458  017470  013701  001450                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5459  017474  012737  000013  001226                MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5460
5461                                        ;;*****************************************************************
5462                                        ;LOOP #1        FORMAT,WRITE,READ
5463
5464  017502                                10$:
5465
5466                                        ;PREPARE THE DEVICE FOR FORMAT OPERATION
5467  017502  004737  040020                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5468  017506  054130                                .WORD   054130  ;TASK DESCRIPTOR
5469  017510  000404                                BR      20$                     ;GO TO 20$ IF NO ERROR
5470  017512  000240                                NOP                     ;RETURN HERE IF ERROR
5471  017514  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5472  017516  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5473  017522                                20$:
5474
5475                                        ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5476  017522  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
5477  017530  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
5478  017536  012737  107546  001404                MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5479  017544  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
5480  017552  012737  010000  001432                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5481  017560  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5482
5483                                        ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5484
5485  017566  004737  040734                        JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5486  017572  000405                                BR      25$                     ;GO TO 25$ IF NO ERROR
5487  017574  104401  067464                        TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5488  017600  104000                                ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5489  017602  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5490  017606                                25$:
5491  017606  012737  071044  001174                MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5492  017614  012737  000001  001176                MOV     #1,$TMP1                ;RANGE OF PATTERN
5493  017622  004737  042640                        JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5494
```

```
5495                                    ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5496  017626  012702  001535                    MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5497  017632  112722  000034                    MOVB    #RMDC,(R2)+
5498  017636  112722  000006                    MOVB    #RMDA,(R2)+
5499  017642  112722  000004                    MOVB    #RMBA,(R2)+
5500  017646  112722  000002                    MOVB    #RMWC,(R2)+
5501  017652  112722  000032                    MOVB    #RMOF,(R2)+
5502  017656  112722  000000                    MOVB    #RMCS1,(R2)+
5503  017662  112712  000200                    MOVB    #200,(R2)               ;TERMINATE TABLE
5504  017666                                30$:
5505
5506                                    ;FORMAT THE DRIVE
5507  017666  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5508  017672  000404                            BR      40$                     ;GO TO 40$ IF NO ERROR
5509  017674  000240                            NOP                     ;RETURN HERE IF ERROR
5510  017676  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5511  017700  000137  020776                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5512  017704                                40$:
5513
5514                                    ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
5515  017704  004737  043452                    JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
5516
5517                                    ;WAIT FOR THE FORMAT TO COMPLETE
5518  017710  004737  044346                    JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
5519
5520                                    ;GO READ STATUS FOR FORMAT OPERATION
5521  017714  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5522  017720  000404                            BR      50$                     ;GO TO 50$ IF NO ERROR
5523  017722  000240                            NOP                     ;RETURN HERE IF ERROR
5524  017724  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5525  017726  000137  020776                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5526  017732                                50$:
5527
5528                                    ;VERIFY NO ERRORS DURING FORMAT
5529  017732  004737  057036                    JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5530  017736  000405                            BR      60$                     ;GO TO 60$ IF NO ERROR
5531  017740  000240                            NOP                     ;RETURN HERE IF ERROR
5532  017742  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5533  017744  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5534  017746  000137  020776                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5535  017752                                60$:
5536
5537                                    ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5538  017752  012737  020012  001122            MOV     #70$,$LPADR
5539  017760  012737  020012  001124            MOV     #70$,$LPERR
5540  017766  013737  001434  021000            MOV     RMDCO,360$
5541  017774  012737  020004  001210            MOV     #65$,$ESCAPE    ;;ESCAPE TO 65$ ON ERROR
5542  020002  000413                            BR      80$                     ;SKIP TO WRITE OPERATION
5543
5544  020004  013737  021000  001434  65$:     MOV     360$,RMDCO              ;RESTORE CYLINDER
5545                                    ;;*****************************************************************
5546                                    ;LOOP #2        WRITE,READ
5547
5548  020012                                70$:
5549
5550
```

```
5551
5552  020012  004737  040020          ;PREPARE DEVICE FOR WRITE OPERATION
5553  020016  054130                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5554  020020  000404                  .WORD   054130  ;TASK DESCRIPTOR
5555  020022  000240                  BR      80$                     ;GO TO 80$ IF NO ERROR
5556  020024  104000                  NOP                     ;RETURN HERE IF ERROR
5557  020026  000137  020776          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5558  020032                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5559                          80$:
5560  020032  013737  001434  021000  MOV     RMDC0,360$              ;SAVE CYLINER
5561  020040  012737  001466  001434  MOV     #822.,RMDC0             ;SEEK TO LAST CYLINER
5562  020046  012737  000005  001400  MOV     #SEEK!GO,RMCS10         ;WRITE SEEK COMMAND
5563  020054  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5564  020060  000404                  BR      90$                     ;GO TO 90$ IF NO ERROR
5565  020062  000240                  NOP                     ;RETURN HERE IF ERROR
5566  020064  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5567  020066  000137  020776          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5568  020072                          90$:
5569
5570                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5571  020072  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5572  020076  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5573  020102  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5574  020106  000404                  BR      100$                    ;GO TO 100$ IF NO ERROR
5575  020110  000240                  NOP                     ;RETURN HERE IF ERROR
5576  020112  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5577  020114  000137  020776          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5578  020120                          100$:
5579  020120  004737  051440          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5580  020124  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
5581  020126  000240                  NOP                     ;RETURN HERE IF ERROR
5582  020130  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5583  020132  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5584  020134  000137  020776          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5585  020140  013737  021000  001434  110$:  MOV     360$,RMDC0              ;RESTORE CYLINDER
5586  020146  012737  000000  001210  MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5587
5588  020154                          120$:
5589
5590                          ;WRITE DATA TO THE DRIVE
5591  020154  012737  177400  001402  MOV     #<^C256.*1>,RMWC0       ;CHANGE WORD COUNT
5592  020162  012737  107552  001404  MOV     #BUF0NE+4,RMBA0         ;CHANGE BUS ADDRESS
5593  020170  012737  000061  001400  MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
5594  020176  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5595  020202  112722  000006          MOVB    #RMDA,(R2)+
5596  020206  112722  000034          MOVB    #RMDC,(R2)+
5597  020212  112722  000032          MOVB    #RMOF,(R2)+
5598  020216  112722  000004          MOVB    #RMBA,(R2)+
5599  020222  112722  000002          MOVB    #RMWC,(R2)+
5600  020226  112722  000000          MOVB    #RMCS1,(R2)+
5601  020232  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
5602
5603  020236  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5604  020242  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
5605  020244  000240                  NOP                     ;RETURN HERE IF ERROR
5606  020246  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
```

```
5607  020250  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5608  020254                              130$:
5609
5610                                      ;WAIT FOR WRITE COMMAND TO COMPLETE
5611  020254  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5612
5613                                      ;GO READ STATUS FOR WRITE COMMAND
5614  020260  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5615  020264  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
5616  020266  000240                        NOP                     ;RETURN HERE IF ERROR
5617  020270  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5618  020272  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5619  020276                              140$:
5620
5621                                      ;CHECK FOR ERRORS DURING WRITE OPERATION
5622  020276  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5623  020302  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5624  020304  000240                        NOP                     ;RETURN HERE IF ERROR
5625  020306  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5626  020310  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5627  020312  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5628  020316                              150$:
5629  020316  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5630  020322  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5631  020324  000240                        NOP                     ;RETURN HERE IF ERROR
5632  020326  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5633  020330  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5634  020332  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5635  020336                              160$:
5636
5637  020336                              170$:
5638  020336  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5639  020342  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
5640  020344  000240                        NOP                     ;RETURN HERE IF ERROR
5641  020346  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5642  020350  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5643  020352  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5644  020356                              180$:
5645
5646                                      ;CHANGE LOOP ADDRESSES
5647  020356  012737  020416  001122        MOV     #190$,$LPADR
5648  020364  012737  020416  001124        MOV     #190$,$LPERR
5649  020372  013737  001434  021000        MOV     RMDCO,360$
5650  020400  012737  020410  001210        MOV     #185$,$ESCAPE   ;;ESCAPE TO 185$ ON ERROR
5651  020406  000413                        BR      200$                    ;SKIP TO NEXT OPERATION
5652  020410  013737  021000  001434  185$: MOV     360$,RMDCO              ;RESTORE CYLINDER
5653
5654                                      ;;*****************************************************************
5655                                      ;LOOP #3        READ
5656
5657  020416                              190$:
5658
5659                                      ;PREPARE DEVICE FOR READ OPERATION
5660  020416  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5661  020422  054130                        .WORD   054130  ;TASK DESCRIPTOR
5662  020424  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
```

```
5663  020426  000240                          NOP                     ;RETURN HERE IF ERROR
5664  020430  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5665  020432  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5666  020436                          200$:
5667
5668  020436  013737  001434  021000          MOV     RMDCO,360$              ;SAVE CYLINDER
5669  020444  012737  001466  001434          MOV     #822.,RMDCO             ;SEEK TO LAST CYLINDER
5670  020452  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;WRITE SEEK COMMAND
5671  020460  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5672  020464  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
5673  020466  000240                          NOP                     ;RETURN HERE IF ERROR
5674  020470  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5675  020472  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5676  020476                          210$:
5677
5678                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5679  020476  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5680  020502  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5681  020506  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5682  020512  000404                          BR      220$                    ;GO TO 220$ IF NO ERROR
5683  020514  000240                          NOP                     ;RETURN HERE IF ERROR
5684  020516  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5685  020520  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5686  020524                          220$:
5687  020524  004737  051440                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5688  020530  000405                          BR      230$                    ;GO TO 230$ IF NO ERROR
5689  020532  000240                          NOP                     ;RETURN HERE IF ERROR
5690  020534  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5691  020536  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5692  020540  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5693  020544  013737  021000  001434  230$:   MOV     360$,RMDCO              ;RESTORE CYLINDER
5694  020552  012737  000000  001210          MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5695
5696  020560                          240$:
5697
5698                                  ;READ DATA FROM DEVICE
5699  020560  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
5700  020566  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;LOAD STARTING BUFFER ADDRESS
5701  020574  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5702  020600  112722  000006                  MOVB    #RMDA,(R2)+
5703  020604  112722  000032                  MOVB    #RMOF,(R2)+
5704  020610  112722  000034                  MOVB    #RMDC,(R2)+
5705  020614  112722  000004                  MOVB    #RMBA,(R2)+
5706  020620  112722  000002                  MOVB    #RMWC,(R2)+
5707  020624  112722  000000                  MOVB    #RMCS1,(R2)+
5708  020630  112712  000200                  MOVB    #200,(R2)
5709
5710  020634  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5711  020640  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
5712  020642  000240                          NOP                     ;RETURN HERE IF ERROR
5713  020644  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5714  020646  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5715  020652                          250$:
5716
5717
5718                                  ;WAIT FOR READ OPERATION TO COMPLETE
```

```
5719  020652  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5720
5721                                ;GO READ STATUS FOR READ OPERATION
5722  020656  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5723  020662  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
5724  020664  000240                        NOP                     ;RETURN HERE IF ERROR
5725  020666  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5726  020670  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5727  020674                        260$:
5728
5729                                ;CHECK FOR ERRORS DURING READ OPERATION
5730  020674  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5731  020700  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
5732  020702  000240                        NOP                     ;RETURN HERE IF ERROR
5733  020704  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5734  020706  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5735  020710  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5736  020714                        270$:
5737  020714  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5738  020720  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
5739  020722  000240                        NOP                     ;RETURN HERE IF ERROR
5740  020724  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5741  020726  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5742  020730  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5743  020734                        280$:
5744
5745  020734                        290$:
5746  020734  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5747  020740  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
5748  020742  000240                        NOP                     ;RETURN HERE IF ERROR
5749  020744  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5750  020746  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5751  020750  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5752  020754                        300$:
5753  020754  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5754  020760  107552                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
5755  020762  110556                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
5756  020764  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
5757  020766  000240                        NOP                     ;RETURN HERE IF ERROR
5758  020770  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5759  020772  000137  020776                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5760  020776                        310$:
5761
5762  020776                        350$:
5763  020776  000401                        BR      370$
5764  021000  000000                360$:   .WORD
5765  021002                        370$:
5766                                ;;*****************************************************************
5767                                ;*TEST 14        WRITE, WRITE CHECK W/ HEAD SWITCHING
5768                                ;;*****************************************************************
5769  021002                        TST14:
5770  021002  000004                        SCOPE                   ;SCOPE CALL
5771  021004  000240                        NOP                     ;START OF TEST
5772  021006  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5773  021012  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5774  021016  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
 5775  021022  012737  000014  001226          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 5776
 5777                                  ;;*****************************************************************
 5778                                  ;LOOP #1        FORMAT,WRITE,READ
 5779
 5780  021030                          10$:
 5781
 5782                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
 5783  021030  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 5784  021034  054130                          .WORD   054130  ;TASK DESCRIPTOR
 5785  021036  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
 5786  021040  000240                          NOP                     ;RETURN HERE IF ERROR
 5787  021042  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 5788  021044  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5789  021050                          20$:
 5790
 5791                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
 5792  021050  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
 5793  021056  012737  000037  001406          MOV     #037,RMDA0              ;TRACK = 0, SECTOR = 31
 5794  021064  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
 5795  021072  012737  176774  001402          MOV     #<^C<2*<256.+2>>+1>,RMWC0               ;WORD COUNT
 5796  021100  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 5797  021106  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
 5798
 5799                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
 5800
 5801  021114  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
 5802  021120  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
 5803  021122  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
 5804  021126  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
 5805  021130  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5806  021134                          25$:
 5807  021134  012737  071044  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
 5808  021142  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
 5809  021150  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
 5810
 5811                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
 5812  021154  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
 5813  021160  112722  000034                  MOVB    #RMDC,(R2)+
 5814  021164  112722  000006                  MOVB    #RMDA,(R2)+
 5815  021170  112722  000004                  MOVB    #RMBA,(R2)+
 5816  021174  112722  000002                  MOVB    #RMWC,(R2)+
 5817  021200  112722  000032                  MOVB    #RMOF,(R2)+
 5818  021204  112722  000000                  MOVB    #RMCS1,(R2)+
 5819  021210  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
 5820  021214                          30$:
 5821
 5822                                  ;FORMAT THE DRIVE
 5823  021214  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 5824  021220  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
 5825  021222  000240                          NOP                     ;RETURN HERE IF ERROR
 5826  021224  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 5827  021226  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5828  021232                          40$:
 5829
 5830                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
```

```
 5831  021232  004737  043452                 JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 5832
 5833                                 ;WAIT FOR THE FORMAT TO COMPLETE
 5834  021236  004737  044346                 JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 5835
 5836                                 ;GO READ STATUS FOR FORMAT OPERATION
 5837  021242  004737  043536                 JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 5838  021246  000404                         BR      50$                     ;GO TO 50$ IF NO ERROR
 5839  021250  000240                         NOP                     ;RETURN HERE IF ERROR
 5840  021252  104000                         ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 5841  021254  000137  022022                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5842  021260                         50$:
 5843
 5844                                 ;VERIFY NO ERRORS DURING FORMAT
 5845  021260  004737  057036                 JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 5846  021264  000405                         BR      60$                     ;GO TO 60$ IF NO ERROR
 5847  021266  000240                         NOP                     ;RETURN HERE IF ERROR
 5848  021270  104000                         ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 5849  021272  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5850  021274  000137  022022                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5851  021300                         60$:
 5852
 5853                                 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 5854  021300  012737  021360  001122         MOV     #70$,$LPADR
 5855  021306  012737  021360  001124         MOV     #70$,$LPERR
 5856                                 ;REGENERATE BUFFER
 5857  021314  012737  177000  001402         MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
 5858  021322  012737  107546  001404         MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRESS
 5859  021330  012737  071106  001174         MOV     #ZEROS,$TMP0            ;STARTING ADDRESS
 5860  021336  012737  000001  001176         MOV     #1,$TMP1                ;RANGE
 5861  021344  012737  000060  001400         MOV     #WD,RMCS1O              ;WRITE DATA
 5862  021352  004737  042640                 JSR     PC,GENBUF               ;GENERATE BUFFER
 5863  021356  000410                         BR      80$                     ;SKIP TO WRITE OPERATION
 5864
 5865                                 ;;*****************************************************************
 5866                                 ;LOOP #2        WRITE,READ
 5867
 5868  021360                         70$:
 5869
 5870
 5871                                 ;PREPARE DEVICE FOR WRITE OPERATION
 5872  021360  004737  040020                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 5873  021364  054130                         .WORD   054130  ;TASK DESCRIPTOR
 5874  021366  000404                         BR      80$                     ;GO TO 80$ IF NO ERROR
 5875  021370  000240                         NOP                     ;RETURN HERE IF ERROR
 5876  021372  104000                         ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 5877  021374  000137  022022                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 5878  021400                         80$:
 5879
 5880
 5881  021400                         120$:
 5882
 5883                                 ;WRITE DATA TO THE DRIVE
 5884  021400  012737  000061  001400         MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
 5885  021406  012702  001535                 MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
 5886  021412  112722  000006                 MOVB    #RMDA,(R2)+
```

```
5887  021416  112722  000034                  MOVB    #RMDC,(R2)+
5888  021422  112722  000032                  MOVB    #RMOF,(R2)+
5889  021426  112722  000004                  MOVB    #RMBA,(R2)+
5890  021432  112722  000002                  MOVB    #RMWC,(R2)+
5891  021436  112722  000000                  MOVB    #RMCS1,(R2)+
5892  021442  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
5893
5894  021446  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5895  021452  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
5896  021454  000240                          NOP                     ;RETURN HERE IF ERROR
5897  021456  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5898  021460  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5899  021464                          130$:
5900
5901                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5902  021464  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5903
5904                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
5905  021470  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5906
5907                                  ;GO READ STATUS FOR WRITE COMMAND
5908  021474  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5909  021500  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
5910  021502  000240                          NOP                     ;RETURN HERE IF ERROR
5911  021504  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5912  021506  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5913  021512                          140$:
5914
5915                                  ;CHECK FOR EPRORS DURING WRITE OPERATION
5916  021512  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5917  021516  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5918  021520  000240                          NOP                     ;RETURN HERE IF ERROR
5919  021522  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5920  021524  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5921  021526  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5922  021532                          150$:
5923  021532  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5924  021536  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5925  021540  000240                          NOP                     ;RETURN HERE IF ERROR
5926  021542  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5927  021544  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5928  021546  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5929  021552                          160$:
5930
5931  021552                          170$:
5932  021552  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5933  021556  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5934  021560  000240                          NOP                     ;RETURN HERE IF ERROR
5935  021562  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5936  021564  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5937  021566  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5938  021572                          180$:
5939
5940                                  ;CHANGE LOOP ADDRESSES
5941  021572  012737  021610  001122          MOV     #190$,$LPADR
5942  021600  012737  021610  001124          MOV     #190$,$LPERR
```

```
5943  021606  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
5944
5945                                  ;;*****************************************************************
5946                                  ;LOOP #3        READ
5947
5948  021610                          190$:
5949
5950                                  ;PREPARE DEVICE FOR READ OPERATION
5951  021610  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5952  021614  054130                        .WORD   054130  ;TASK DESCRIPTOR
5953  021616  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5954  021620  000240                        NOP                     ;RETURN HERE IF ERROR
5955  021622  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5956  021624  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5957  021630                          200$:
5958
5959  021630                          240$:
5960
5961                                  ;READ DATA FROM DEVICE
5962  021630  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5963  021636  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5964  021642  112722  000006                MOVB    #RMDA,(R2)+
5965  021646  112722  000032                MOVB    #RMOF,(R2)+
5966  021652  112722  000034                MOVB    #RMDC,(R2)+
5967  021656  112722  000004                MOVB    #RMBA,(R2)+
5968  021662  112722  000002                MOVB    #RMWC,(R2)+
5969  021666  112722  000000                MOVB    #RMCS1,(R2)+
5970  021672  112712  000200                MOVB    #200,(R2)
5971
5972  021676  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5973  021702  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
5974  021704  000240                        NOP                     ;RETURN HERE IF ERROR
5975  021706  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5976  021710  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5977  021714                          250$:
5978
5979                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5980  021714  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5981
5982                                  ;WAIT FOR READ OPERATION TO COMPLETE
5983  021720  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5984
5985                                  ;GO READ STATUS FOR READ OPERATION
5986  021724  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5987  021730  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
5988  021732  000240                        NOP                     ;RETURN HERE IF ERROR
5989  021734  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5990  021736  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5991  021742                          260$:
5992
5993                                  ;CHECK FOR ERRORS DURING READ OPERATION
5994  021742  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5995  021746  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
5996  021750  000240                        NOP                     ;RETURN HERE IF ERROR
5997  021752  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5998  021754  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
 5999  021756  000137  022022                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6000  021762                                270$:
 6001  021762  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 6002  021766  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
 6003  021770  000240                            NOP                     ;RETURN HERE IF ERROR
 6004  021772  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 6005  021774  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6006  021776  000137  022022                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6007  022002                                280$:
 6008
 6009  022002                                290$:
 6010  022002  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 6011  022006  000405                            BR      300$                    ;GO TO 300$ IF NO ERROR
 6012  022010  000240                            NOP                     ;RETURN HERE IF ERROR
 6013  022012  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 6014  022014  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6015  022016  000137  022022                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6016  022022                                300$:
 6017
 6018  022022                                350$:
 6019                                          ;;*****************************************************************
 6020                                          ;*TEST 15        WRITE, WRITE CHECK W/ MID-TRANSFER SEEK
 6021                                          ;;*****************************************************************
 6022  022022                                TST15:
 6023  022022  000004                            SCOPE                   ;SCOPE CALL
 6024  022024  000240                            NOP                     ;START OF TEST
 6025  022026  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 6026  022032  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 6027  022036  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 6028  022042  012737  000015  001226            MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 6029
 6030                                          ;;*****************************************************************
 6031                                          ;LOOP #1        FORMAT,WRITE,READ
 6032
 6033  022050                                10$:
 6034
 6035                                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
 6036  022050  004737  040020                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 6037  022054  054130                                .WORD   054130  ;TASK DESCRIPTOR
 6038  022056  000404                                BR      20$                     ;GO TO 20$ IF NO ERROR
 6039  022060  000240                                NOP                     ;RETURN HERE IF ERROR
 6040  022062  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 6041  022064  000137  023042                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6042  022070                                20$:
 6043
 6044                                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
 6045  022070  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
 6046  022076  012737  002037  001406                MOV     #2037,RMDA0             ;TRACK = 4, SECTOR = 31
 6047  022104  012737  107546  001404                MOV     #BUFONE,RMBA0           ;BUS ADDRESS
 6048  022112  012737  176774  001402                MOV     #<^C<2*<256.+2>>+1>,RMWC0               ;WORD COUNT
 6049  022120  012737  010000  001432                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 6050  022126  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
 6051
 6052                                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
 6053
 6054  022134  004737  040734                        JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
```

```
 6055  022140  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
 6056  022142  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
 6057  022146  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
 6058  022150  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6059  022154                          25$:
 6060  022154  012737  071106  001174          MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
 6061  022162  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
 6062  022170  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
 6063
 6064                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
 6065  022174  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
 6066  022200  112722  000034                  MOVB    #RMDC,(R2)+
 6067  022204  112722  000006                  MOVB    #RMDA,(R2)+
 6068  022210  112722  000004                  MOVB    #RMBA,(R2)+
 6069  022214  112722  000002                  MOVB    #RMWC,(R2)+
 6070  022220  112722  000032                  MOVB    #RMOF,(R2)+
 6071  022224  112722  000000                  MOVB    #RMCS1,(R2)+
 6072  022230  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
 6073  022234                          30$:
 6074
 6075                          ;FORMAT THE DRIVE
 6076  022234  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 6077  022240  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
 6078  022242  000240                          NOP                     ;RETURN HERE IF ERROR
 6079  022244  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 6080  022246  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6081  022252                          40$:
 6082
 6083                          ;SETUP GET RFGISTER INDEX TABLE FOR READING STATUS
 6084  022252  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 6085
 6086                          ;WAIT FOR THE FORMAT TO COMPLETE
 6087  022256  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 6088
 6089                          ;GO READ STATUS FOR FORMAT OPERATION
 6090  022262  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 6091  022266  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
 6092  022270  000240                          NOP                     ;RETURN HERE IF ERROR
 6093  022272  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 6094  022274  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6095  022300                          50$:
 6096
 6097                          ;VERIFY NO ERRORS DURING FORMAT
 6098  022300  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 6099  022304  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
 6100  022306  000240                          NOP                     ;RETURN HERE IF ERROR
 6101  022310  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 6102  022312  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6103  022314  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6104  022320                          60$:
 6105
 6106                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 6107  022320  012737  022400  001122          MOV     #70$,$LPADR
 6108  022326  012737  022400  001124          MOV     #70$,$LPERR
 6109                          ;REGENERATE BUFFER
 6110  022334  012737  000060  001400          MOV     #WD,RMCS10              ;WRITE DATA
```

```
6111  022342  012737  177000  001402          MOV     #<^C<2*256.>+1>,RMWLC   ;CHANGE WORD COUNT
6112  022350  012737  107546  001404          MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRSS
6113  022356  012737  071044  001174          MOV     #ONES,$TMP0             ;PATTERN ADDRESS
6114  022364  012737  000001  001176          MOV     #1,$TMP1                ;PATTERN RANGE
6115  022372  004737  042640                  JSR     PC,GENBUF
6116  022376  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
6117
6118                                  ;;*****************************************************************
6119                                  ;LOOP #2        WRITE,READ
6120
6121  022400                          70$:
6122
6123
6124                                  ;PREPARE DEVICE FOR WRITE OPERATION
6125  022400  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6126  022404  054130                          .WORD   054130  ;TASK DESCRIPTOR
6127  022406  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
6128  022410  000240                          NOP                     ;RETURN HERE IF ERROR
6129  022412  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6130  022414  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6131  022420                          80$:
6132
6133
6134  022420                          120$:
6135
6136                                  ;WRITE DATA TO THE DRIVE
6137  022420  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
6138  022426  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6139  022432  112722  000006                  MOVB    #RMDA,(R2)+
6140  022436  112722  000034                  MOVB    #RMDC,(R2)+
6141  022442  112722  000032                  MOVB    #RMOF,(R2)+
6142  022446  112722  000004                  MOVB    #RMBA,(R2)+
6143  022452  112722  000002                  MOVB    #RMWC,(R2)+
6144  022456  112722  000000                  MOVB    #RMCS1,(R2)+
6145  022462  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
6146
6147  022466  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6148  022472  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
6149  022474  000240                          NOP                     ;RETURN HERE IF ERROR
6150  022476  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6151  022500  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6152  022504                          130$:
6153
6154                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6155  022504  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6156
6157                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
6158  022510  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6159
6160                                  ;GO READ STATUS FOR WRITE COMMAND
6161  022514  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6162  022520  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
6163  022522  000240                          NOP                     ;RETURN HERE IF ERROR
6164  022524  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6165  022526  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6166  022532                          140$:
```

```
6167
6168                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
6169  022532  004737  044532            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6170  022536  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
6171  022540  000240                    NOP                     ;RETURN HERE IF ERROR
6172  022542  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6173  022544  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6174  022546  000137  023042            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6175  022552                            150$:
6176  022552  004737  057036            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6177  022556  000405                    BR      160$                    ;GO TO 160$ IF NO ERROR
6178  022560  000240                    NOP                     ;RETURN HERE IF ERROR
6179  022562  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6180  022564  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6181  022566  000137  023042            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6182  022572                            160$:
6183
6184  022572                            170$:
6185  022572  004737  045364            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6186  022576  000405                    BR      180$                    ;GO TO 180$ IF NO ERROR
6187  022600  000240                    NOP                     ;RETURN HERE IF ERROR
6188  022602  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6189  022604  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6190  022606  000137  023042            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6191  022612                            180$:
6192
6193                                    ;CHANGE LOOP ADDRESSES
6194  022612  012737  022630  001122    MOV     #190$,$LPADR
6195  022620  012737  022630  001124    MOV     #190$,$LPERR
6196  022626  000410                    BR      200$                    ;SKIP TO NEXT OPERATION
6197
6198                                    ;;*****************************************************************
6199                                    ;LOOP #3        READ
6200
6201  022630                            190$:
6202
6203                                    ;PREPARE DEVICE FOR READ OPERATION
6204  022630  004737  040020            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6205  022634  054130                    .WORD   054130  ;TASK DESCRIPTOR
6206  022636  000404                    BR      200$                    ;GO TO 200$ IF NO ERROR
6207  022640  000240                    NOP                     ;RETURN HERE IF ERROR
6208  022642  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6209  022644  000137  023042            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6210  022650                            200$:
6211
6212  022650                            240$:
6213
6214                                    ;READ DATA FROM DEVICE
6215  022650  012737  000051  001400    MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
6216  022656  012702  001535            MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6217  022662  112722  000006            MOVB    #RMDA,(R2)+
6218  022666  112722  000032            MOVB    #RMOF,(R2)+
6219  022672  112722  000034            MOVB    #RMDC,(R2)+
6220  022676  112722  000004            MOVB    #RMBA,(R2)+
6221  022702  112722  000002            MOVB    #RMWC,(R2)+
6222  022706  112722  000000            MOVB    #RMCS1,(R2)+
```

```
6223  022712  112712  000200                  MOVB    #200,(R2)
6224
6225  022716  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6226  022722  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6227  022724  000240                          NOP                     ;RETURN HERE IF ERROR
6228  022726  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6229  022730  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6230  022734                          250$:
6231
6232                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6233  022734  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6234
6235                                  ;WAIT FOR READ OPERATION TO COMPLETE
6236  022740  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6237
6238                                  ;GO READ STATUS FOR READ OPERATION
6239  022744  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6240  022750  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6241  022752  000240                          NOP                     ;RETURN HERE IF ERROR
6242  022754  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6243  022756  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6244  022762                          260$:
6245
6246                                  ;CHECK FOR ERRORS DURING READ OPERATION
6247  022762  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6248  022766  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
6249  022770  000240                          NOP                     ;RETURN HERE IF ERROR
6250  022772  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6251  022774  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6252  022776  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6253  023002                          270$:
6254  023002  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6255  023006  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
6256  023010  000240                          NOP                     ;RETURN HERE IF ERROR
6257  023012  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6258  023014  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6259  023016  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6260  023022                          280$:
6261
6262  023022                          290$:
6263  023022  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6264  023026  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
6265  023030  000240                          NOP                     ;RETURN HERE IF ERROR
6266  023032  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6267  023034  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6268  023036  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6269  023042                          300$:
6270
6271  023042                          350$:
6272                                  ;;**********************************************************************
6273                                  ;*TEST 16        WRITE, READ W/ HCE ERROR
6274                                  ;;**********************************************************************
6275  023042                          TST16:
6276  023042  000004                          SCOPE                   ;SCOPE CALL
6277  023044  000240                          NOP                     ;START OF TEST
6278  023046  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
6279  023052  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6280  023056  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6281  023062  012737  000016  001226        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6282
6283                                ;;*****************************************************************
6284                                ;LOOP #1        FORMAT,WRITE,READ
6285
6286  023070  012704  000000                MOV     #0,R4                   ;R4=HEADER WORD
6287                                ;;*****************************************************************
6288                                ;NOTE:  BSE NOT TESTED
6289  023074  012703  020000        10$:    MOV     #BIT13,R3               ;R3=HCE BIT
6290  023100                        15$:
6291
6292                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
6293  023100  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6294  023104  054130                        .WORD   054130  ;TASK DESCRIPTOR
6295  023106  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
6296  023110  000240                        NOP                     ;RETURN HERE IF ERROR
6297  023112  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6298  023114  000137  024724                JMP     370$                    ;GO TO 370$ IF ERROR WAS FOUND
6299  023120                        20$:
6300
6301                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
6302  023120  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
6303  023126  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
6304  023134  012737  107546  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
6305  023142  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
6306  023150  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6307  023156  012737  000063  001400        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
6308
6309                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
6310
6311  023164  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
6312  023170  000405                        BR      25$                     ;GO TO 25$ IF NO ERROR
6313  023172  104401  067464                TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
6314  023176  104000                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
6315  023200  000137  024536                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6316  023204                        25$:
6317  023204  012737  071106  001174        MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
6318  023212  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
6319  023220  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
6320
6321                                ;CHANGE HEADER WORD TO FORCE HCE DURING WRITE & READ
6322  023224  030364  107546                BIT     R3,BUFONE(R4)           ;SET OR RESET FOR HCE??
6323  023230  001403                        BEQ     26$                     ;
6324  023232  040364  107546                BIC     R3,BUFONE(R4)           ;RESET BIT FOR HCE
6325  023236  000402                        BR      27$
6326  023240  050364  107546        26$:    BIS     R3,BUFONE(R4)           ;SET FOR HCE
6327  023244                        27$:
6328                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
6329  023244  012702  001535                MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
6330  023250  112722  000034                MOVB    #RMDC,(R2)+
6331  023254  112722  000006                MOVB    #RMDA,(R2)+
6332  023260  112722  000004                MOVB    #RMBA,(R2)+
6333  023264  112722  000002                MOVB    #RMWC,(R2)+
6334  023270  112722  000032                MOVB    #RMOF,(R2)+
```

```
6335  023274  112722  000000          MOVB    #RMCS1,(R2)+
6336  023300  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
6337  023304                        30$:
6338
6339                                ;FORMAT THE DRIVE
6340  023304  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6341  023310  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
6342  023312  000240                  NOP                     ;RETURN HERE IF ERROR
6343  023314  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6344  023316  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6345  023322                        40$:
6346
6347                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
6348  023322  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6349
6350                                ;WAIT FOR THE FORMAT TO COMPLETE
6351  023326  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6352
6353                                ;GO READ STATUS FOR FORMAT OPERATION
6354  023332  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6355  023336  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
6356  023340  000240                  NOP                     ;RETURN HERE IF ERROR
6357  023342  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6358  023344  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6359  023350                        50$:
6360
6361                                ;VERIFY NO ERRORS DURING FORMAT
6362  023350  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6363  023354  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
6364  023356  000240                  NOP                     ;RETURN HERE IF ERROR
6365  023360  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6366  023362  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6367  023364  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6368  023370                        60$:
6369
6370                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
6371  023370  012737  023406  001122    MOV     #70$,$LPADR
6372  023376  012737  023406  001124    MOV     #70$,$LPERR
6373  023404  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
6374
6375                                ;;*****************************************************************
6376                                ;LOOP #2        WRITE,READ
6377
6378  023406                        70$:
6379
6380
6381                                ;PREPARE DEVICE FOR WRITE OPERATION
6382  023406  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6383  023412  054130                  .WORD   054130  ;TASK DESCRIPTOR
6384  023414  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
6385  023416  000240                  NOP                     ;RETURN HERE IF ERROR
6386  023420  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6387  023422  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6388  023426                        80$:
6389
6390
```

```
6391  023426                                  120$:
6392
6393                                          ;WRITE DATA TO THE DRIVE
6394  023426  012737  107552  001404          MOV     #BUFONE+4,RMBA0         ;CHANGE BUS ADDRESS
6395  023434  012737  177400  001402          MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
6396  023442  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
6397  023450  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6398  023454  112722  000006                  MOVB    #RMDA,(R2)+
6399  023460  112722  000034                  MOVB    #RMDC,(R2)+
6400  023464  112722  000032                  MOVB    #RMOF,(R2)+
6401  023470  112722  000004                  MOVB    #RMBA,(R2)+
6402  023474  112722  000002                  MOVB    #RMWC,(R2)+
6403  023500  112722  000000                  MOVB    #RMCS1,(R2)+
6404  023504  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
6405
6406  023510  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6407  023514  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
6408  023516  000240                          NOP                     ;RETURN HERE IF ERROR
6409  023520  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6410  023522  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6411  023526                                  130$:
6412
6413                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6414  023526  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
6415
6416                                          ;WAIT FOR WRITE COMMAND TO COMPLETE
6417  023532  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
6418
6419                                          ;GO READ STATUS FOR WRITE COMMAND
6420  023536  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6421  023542  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
6422  023544  000240                          NOP                     ;RETURN HERE IF ERROR
6423  023546  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6424  023550  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6425  023554                                  140$:
6426
6427                                          ;CHECK FOR ERRORS DURING WRITE OPERATION
6428  023554  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6429  023560  000405                          BR      145$                    ;GO TO 145$ IF NO ERROR
6430  023562  000240                          NOP                     ;RETURN HERE IF ERROR
6431  023564  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6432  023566  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6433  023570  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6434  023574                                  145$:
6435
6436                                          ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
6437  023574  005704                          TST     R4                      ;IS THIS THE FIRST HEADER WORD ?
6438  023576  001006                          BNE     146$                    ;NO !!
6439  023600  032703  010000                  BIT     #FMT16,R3               ;IS THIS A FORMAT ERROR ??
6440  023604  001034                          BNE     155$                    ;YES !!
6441  023606  032703  140000                  BIT     #MSE!USE,R3             ;IS THIS A BAD SECTOR ERROR ??
6442  023612  001056                          BNE     165$                    ;YES !!
6443  023614                                  146$:
6444
6445                                          ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
6446  023614  032737  000200  001344          BIT     #HCE,RMER11             ;WAS HCE DETECTED ??
```

```
6447  023622  001077                          BNE     175$                    ;YES !!
6448                                  ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6449  023624  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6450  023630  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6451  023632  000240                          NOP                     ;RETURN HERE IF ERROR
6452  023634  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6453  023636  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6454  023640  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6455  023644                          150$:
6456  023644  013737  001344  001142          MOV     RMER11,$BDDAT           ;RECEIVED STATUS
6457  023652  012737  000200  001140          MOV     #HCE,$GDDAT             ;EXPECTED STATUS
6458  023660  010437  001174                  MOV     R4,$TMP0        ;R4 = HEADER WORD NUMBER
6459  023664  010337  001176                  MOV     R3,$TMP1        ;R3 = BIT POSITION
6460  023670  104344                          ERROR   344                     ;HCE NOT DETECTED
6461  023672  000137  024536                  JMP     350$
6462  023676                          155$:
6463
6464                                  ;VERIFY THAT A FORMAT ERROR WAS DETECTED
6465  023676  032737  000020  001344          BIT     #FER,RMER11             ;WAS FER DETECTED ??
6466  023704  001046                          BNE     175$                    ;YES !!
6467                                  ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6468  023706  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6469  023712  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6470  023714  000240                          NOP                     ;RETURN HERE IF ERROR
6471  023716  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6472  023720  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6473  023722  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6474  023726                          160$:
6475  023726  012737  000020  001140          MOV     #FER,$GDDAT             ;EXPECTED STATUS
6476  023734  013737  001344  001142          MOV     RMER11,$BDDAT           ;RECEIVED STATUS
6477  023742  104343                          ERROR   343                     ;FER NOT DETECTED
6478  023744  000137  024536                  JMP     350$
6479  023750                          165$:
6480
6481                                  ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
6482  023750  032737  100000  001372          BIT     #BSE,RMER21             ;WAS BSE DETECTED ??
6483  023756  001021                          BNE     175$                    ;YES !!
6484                                  ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6485  023760  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6486  023764  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6487  023766  000240                          NOP                     ;RETURN HERE IF ERROR
6488  023770  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6489  023772  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6490  023774  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6491  024000                          170$:
6492  024000  013737  001372  001142          MOV     RMER21,$BDDAT           ;RECEIVED STATUS
6493  024006  012737  100000  001140          MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
6494  024014  104345                          ERROR   345                     ;BSE NOTDETECTED
6495  024016  000137  024536                  JMP     350$
6496  024022                          175$:
6497  024022  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6498  024026  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
6499  024030  000240                          NOP                     ;RETURN HERE IF ERROR
6500  024032  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6501  024034  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6502  024036  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
6503  024042                          180$:
6504
6505                                  ;CHANGE LOOP ADDRESSES
6506  024042  012737  024056  001122          MOV     #190$,$LPADR
6507  024050  012737  024056  001124          MOV     #190$,$LPERR
6508
6509                                  ;;*****************************************************************
6510                                  ;LOOP #3        READ
6511
6512  024056                          190$:
6513
6514                                  ;PREPARE DEVICE FOR READ OPERATION
6515  024056  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6516  024062  054130                          .WORD   054130  ;TASK DESCRIPTOR
6517  024064  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
6518  024066  000240                          NOP                     ;RETURN HERE IF ERROR
6519  024070  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6520  024072  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6521  024076                          200$:
6522
6523  024076                          240$:
6524
6525                                  ;READ DATA FROM DEVICE
6526  024076  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;CHANGE BUS ADDRESS
6527  024104  012737  177400  001402          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
6528  024112  012737  000071  001400          MOV     #RD!GO,RMCS1O           ;READ DATA COMMAND
6529  024120  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6530  024124  112722  000006                  MOVB    #RMDA,(R2)+
6531  024130  112722  000032                  MOVB    #RMOF,(R2)+
6532  024134  112722  000034                  MOVB    #RMDC,(R2)+
6533  024140  112722  000004                  MOVB    #RMBA,(R2)+
6534  024144  112722  000002                  MOVB    #RMWC,(R2)+
6535  024150  112722  000000                  MOVB    #RMCS1,(R2)+
6536  024154  112712  000200                  MOVB    #200,(R2)
6537
6538  024160  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6539  024164  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6540  024166  000240                          NOP                     ;RETURN HERE IF ERROR
6541  024170  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6542  024172  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6543  024176                          250$:
6544
6545                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6546  024176  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6547
6548                                  ;WAIT FOR READ OPERATION TO COMPLETE
6549  024202  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6550
6551                                  ;GO READ STATUS FOR READ OPERATION
6552  024206  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6553  024212  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6554  024214  000240                          NOP                     ;RETURN HERE IF ERROR
6555  024216  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6556  024220  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6557  024224                          260$:
6558
```

```
6559                                    ;CHECK FOR ERRORS DURING READ OPERATION
6560  024224  004737  044532            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6561  024230  000405                    BR      265$                    ;GO TO 265$ IF NO ERROR
6562  024232  000240                    NOP                     ;RETURN HERE IF ERROR
6563  024234  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6564  024236  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6565  024240  000137  024536            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6566  024244                          265$:
6567                                    ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
6568  024244  005704                    TST     R4                      ;IS THIS THE FIRST HEADER WORD ?
6569  024246  001006                    BNE     266$                    ;NO !!
6570  024250  032703  010000            BIT     #FMT16,R3               ;IS THIS A FORMAT ERROR ??
6571  024254  001034                    BNE     275$                    ;YES !!
6572  024256  032703  140000            BIT     #MSE!USE,R3             ;IS THIS A BAD SECTOR ERROR ??
6573  024262  001056                    BNE     285$                    ;YES !!
6574  024264                          266$:
6575
6576                                    ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
6577  024264  032737  000200  001344    BIT     #HCE,RMER11             ;WAS HCE DETECTED ??
6578  024272  001077                    BNE     295$                    ;YES !!
6579                                    ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6580  024274  004737  057036            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6581  024300  000405                    BR      270$                    ;GO TO 270$ IF NO ERROR
6582  024302  000240                    NOP                     ;RETURN HERE IF ERROR
6583  024304  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6584  024306  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6585  024310  000137  024536            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6586  024314                          270$:
6587  024314  013737  001344  001142    MOV     RMER11,$BDDAT           ;RECEIVED STATUS
6588  024322  012737  000200  001140    MOV     #HCE,$GDDAT             ;EXPECTED STATUS
6589  024330  010437  001174            MOV     R4,$TMP0        ;R4 = HEADER WORD NUMBER
6590  024334  010337  001176            MOV     R3,$TMP1        ;R3 = BIT POSITION
6591  024340  104344                    ERROR   344                     ;HCE NOT DETECTED
6592  024342  000137  024536            JMP     350$
6593  024346                          275$:
6594
6595                                    ;VERIFY THAT A FORMAT ERROR WAS DETECTED
6596  024346  032737  000020  001344    BIT     #FER,RMER11             ;WAS FER DETECTED ??
6597  024354  001046                    BNE     295$                    ;YES !!
6598                                    ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6599  024356  004737  057036            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6600  024362  000405                    BR      280$                    ;GO TO 280$ IF NO ERROR
6601  024364  000240                    NOP                     ;RETURN HERE IF ERROR
6602  024366  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6603  024370  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6604  024372  000137  024536            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6605  024376                          280$:
6606  024376  012737  000020  001140    MOV     #FER,$GDDAT             ;EXPECTED STATUS
6607  024404  013737  001344  001142    MOV     RMER11,$BDDAT           ;RECEIVED STATUS
6608  024412  104343                    ERROR   343                     ;FER NOT DETECTED
6609  024414  000137  024536            JMP     350$
6610  024420                          285$:
6611
6612                                    ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
6613  024420  032737  100000  001372    BIT     #BSE,RMER21             ;WAS BSE DETECTED ??
6614  024426  001021                    BNE     295$                    ;YES !!
```

```
6615
6616  024430  004737  057036          ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6617  024434  000405                      JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6618  024436  000240                      BR      290$                    ;GO TO 290$ IF NO ERROR
6619  024440  104000                      NOP                     ;RETURN HERE IF ERROR
6620  024442  004736                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6621  024444  000137  024536              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6622  024450                              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6623  024450  013737  001372  001142  290$:
6624  024456  012737  100000  001140      MOV     RMER21,$BDDAT           ;RECEIVED STATUS
6625  024464  104345                      MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
6626  024466  000137  000350              ERROR   345                     ;BSE NOTDETECTED
6627  024472                              JMP     350
6628  024472  004737  045364  295$:
6629  024476  000405                      JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6630  024500  000240                      BR      300$                    ;GO TO 300$ IF NO ERROR
6631  024502  104000                      NOP                     ;RETURN HERE IF ERROR
6632  024504  004736                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6633  024506  000137  024536              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6634  024512                              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6635                                  300$:
6636  024512  006203                      ASR     R3                      ;SHIFT TO NEXT BIT
6637  024514  001006                      BNE     310$                    ;REPEAT IF NOT DONE
6638  024516  005704                      TST     R4                      ;SECOND HEADER WORD DONE??
6639  024520  001007                      BNE     355$                    ;YES!!
6640  024522  012704  000002              MOV     #2,R4                   ;SETUP FOR SECOND HEADER WORD
6641                                  ;;*****************************************************************
6642                                  ;NOTE:  BSE NOT TESTED
6643  024526  012703  020000              MOV     #BIT13,R3
6644  024532                          310$:
6645  024532  000137  023100              JMP     15$
6646  024536                          350$:
6647  024536  000472                      BR      370$            ;EXIT IF ERROR
6648  024540                          355$:
6649  024540  013737  001476  001434      MOV     ASNDC,RMDC0             ;
6650  024546  013737  001500  001406      MOV     ASNDA,RMDA0             ;
6651  024554  012737  107546  001404      MOV     #BUFONE,RMBA0           ;BUFFER ADDRESS,REFORMAT THE SECTOR
6652  024562  012737  177776  001402      MOV     #-2,RMWC0               ;ONLY TWO HEAD WORDS
6653  024570  012737  010000  001432      MOV     #FMT16,RMOF0            ;ALWAYS IN 16 BITS MODE
6654  024576  012737  000062  001400      MOV     #WH,RMCS10              ;WRITE HEAD AND DATA COMMAND
6655  024604  004737  042640              JSR     PC,GENBUF               ;SET UP THE BUFFER
6656
6657  024610  004737  040020              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6658  024614  054130                      .WORD   054130  ;TASK DESCRIPTOR
6659  024616  000404                      BR      360$                    ;GO TO 360$ IF NO ERROR
6660  024620  000240                      NOP                     ;RETURN HERE IF ERROR
6661  024622  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6662  024624  000137  024630              JMP     360$                    ;GO TO 360$ IF ERROR WAS FOUND
6663  024630                          360$:
6664  024630  012737  000063  001400      MOV     #WH!GO,RMCS10           ;FORMAT THE SECTOR
6665  024636  012702  001535              MOV     #PUTINX,R2              ;SET UP THE REGS
6666  024642  112722  000006              MOVB    #RMDA,(R2)+             ;
6667  024646  112722  000032              MOVB    #RMOF,(R2)+             ;
6668  024652  112722  000034              MOVB    #RMDC,(R2)+             ;
6669  024656  112722  000004              MOVB    #RMBA,(R2)+             ;
6670  024662  112722  000002              MOVB    #RMWC,(R2)+             ;
```

```
6671  024666  112722  000000                  MOVB    #RMCS1,(R2)+            ;
6672  024672  112722  000200                  MOVB    #200,(R2)+              ;
6673  024676  004737  044006                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6674  024702  000404                          BR      365$    ;GO TO 365$ IF NO ERROR
6675  024704  000240                          NOP                     ;RETURN HERE IF ERROR
6676  024706  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6677  024710  000137  024724                  JMP     370$    ;GO TO 370$ IF ERROR WAS FOUND
6678  024714  000240                  365$:   NOP
6679  024716  004737  044346                  JSR     PC,TIMOUT               ;WAIT FOR FINISH
6680  024722  000240                          NOP
6681  024724  012737  023074  001122  370$:   MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
6682  024732  012737  023074  001124          MOV     #10$,$LPERR
6683                                  ;;*****************************************************************
6684                                  ;*TEST 17        WRITE, READ W/ HCI
6685                                  ;;*****************************************************************
6686  024740                          TST17:
6687  024740  000004                          SCOPE                   ;SCOPE CALL
6688  024742  000240                          NOP                     ;START OF TEST
6689  024744  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6690  024750  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6691  024754  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6692  024760  012737  000017  001226          MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6693
6694                                  ;;*****************************************************************
6695                                  ;LOOP #1        FORMAT,WRITE,READ
6696
6697  024766  012704  000000          10$:    MOV     #0,R4                   ;HEADER WORD
6698  024772  012703  000001                  MOV     #1,R3                   ;HCE BIT
6699
6700  024776                          15$:
6701                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
6702  024776  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6703  025002  054130                          .WORD   054130  ;TASK DESCRIPTOR
6704  025004  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
6705  025006  000240                          NOP                     ;RETURN HERE IF ERROR
6706  025010  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6707  025012  000137  026400                  JMP     370$                    ;GO TO 370$ IF ERROR WAS FOUND
6708  025016                          20$:
6709
6710                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
6711  025016  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
6712  025024  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
6713  025032  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
6714  025040  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
6715  025046  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6716  025054  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
6717
6718                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
6719
6720  025062  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
6721  025066  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
6722  025070  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
6723  025074  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
6724  025076  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6725  025102                          25$:
6726  025102  012737  071044  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
```

```
6727  025110  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
6728  025116  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
6729
6730                                  ;CHANGE HEADER WORD TO FORCE ERROR DURING WRITE
6731  025122  030364  107546                  BIT     R3,BUFONE(R4)           ;SET OR RESET BIT??
6732  025126  001403                          BEQ     27$
6733  025130  040364  107546                  BIC     R3,BUFONE(R4)           ;RESET BIT
6734  025134  000402                          BR      28$
6735  025136  050364  107546          27$:    BIS     R3,BUFONE(R4)           ;SET BIT
6736  025142                          28$:
6737                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
6738  025142  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
6739  025146  112722  000034                  MOVB    #RMDC,(R2)+
6740  025152  112722  000006                  MOVB    #RMDA,(R2)+
6741  025156  112722  000004                  MOVB    #RMBA,(R2)+
6742  025162  112722  000002                  MOVB    #RMWC,(R2)+
6743  025166  112722  000032                  MOVB    #RMOF,(R2)+
6744  025172  112722  000000                  MOVB    #RMCS1,(R2)+
6745  025176  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
6746  025202                          30$:
6747
6748                                  ;FORMAT THE DRIVE
6749  025202  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6750  025206  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
6751  025210  000240                          NOP                     ;RETURN HERE IF ERROR
6752  025212  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6753  025214  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6754  025220                          40$:
6755
6756                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
6757  025220  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6758
6759                                  ;WAIT FOR THE FORMAT TO COMPLETE
6760  025224  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6761
6762                                  ;GO READ STATUS FOR FORMAT OPERATION
6763  025230  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6764  025234  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
6765  025236  000240                          NOP                     ;RETURN HERE IF ERROR
6766  025240  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6767  025242  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6768  025246                          50$:
6769
6770                                  ;VERIFY NO ERRORS DURING FORMAT
6771  025246  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6772  025252  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
6773  025254  000240                          NOP                     ;RETURN HERE IF ERROR
6774  025256  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6775  025260  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6776  025262  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6777  025266                          60$:
6778
6779                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
6780  025266  012737  025304  001122          MOV     #70$,$LPADR
6781  025274  012737  025304  001124          MOV     #70$,$LPERR
6782  025302  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
```

```
6783
6784                                    ;;*****************************************************************
6785                                    ;LOOP #2        WRITE,READ
6786
6787  025304                            70$:
6788
6789
6790                                    ;PREPARE DEVICE FOR WRITE OPERATION
6791  025304  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6792  025310  054130                          .WORD   054130  ;TASK DESCRIPTOR
6793  025312  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
6794  025314  000240                          NOP                     ;RETURN HERE IF ERROR
6795  025316  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6796  025320  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6797  025324                            80$:
6798
6799
6800  025324                            120$:
6801
6802                                    ;WRITE DATA TO THE DRIVE
6803  025324  012737  012000  001432          MOV     #HCI!FMT16,RMOFO        ;INHIBIT HEADER COMPARE
6804  025332  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
6805  025340  012737  107552  001404          MOV     #BUFONE+4,RMBAO         ;LOAD STARTING BUFFER ADDRESS
6806  025346  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6807  025352  112722  000006                  MOVB    #RMDA,(R2)+
6808  025356  112722  000034                  MOVB    #RMDC,(R2)+
6809  025362  112722  000032                  MOVB    #RMOF,(R2)+
6810  025366  112722  000004                  MOVB    #RMBA,(R2)+
6811  025372  112722  000002                  MOVB    #RMWC,(R2)+
6812  025376  112722  000000                  MOVB    #RMCS1,(R2)+
6813  025402  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
6814
6815  025406  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6816  025412  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
6817  025414  000240                          NOP                     ;RETURN HERE IF ERROR
6818  025416  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6819  025420  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6820  025424                            130$:
6821
6822                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6823  025424  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6824
6825                                    ;WAIT FOR WRITE COMMAND TO COMPLETE
6826  025430  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6827
6828                                    ;GO READ STATUS FOR WRITE COMMAND
6829  025434  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6830  025440  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
6831  025442  000240                          NOP                     ;RETURN HERE IF ERROR
6832  025444  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6833  025446  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6834  025452                            140$:
6835
6836                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
6837  025452  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6838  025456  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
6839  025460  000240                          NOP                     ;RETURN HERE IF ERROR
6840  025462  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6841  025464  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6842  025466  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6843  025472                          150$:
6844  025472  032737  000220  001344          BIT     #HCE!FER,RMER1I         ;ANY ERROR??
6845  025500  001407                          BEQ     151$
6846  025502  013737  001344  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6847  025510  042737  177557  001142          BIC     #^C<HCE!FER>,$BDDAT     ;CLEAR DONT CARES
6848  025516  000412                          BR      152$
6849  025520  032737  100000  001372  151$:   BIT     #BSE,RMER2I             ;ANY BAD SECTOR ERRROR ??
6850  025526  001414                          BEQ     155$                    ;NO !!
6851  025530  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6852  025536  042737  077777  001142          BIC     #^CBSE,$BDDAT           ;CLEAR DONT CARES
6853  025544  012737  000000  001140  152$:   MOV     #0,$GDDAT               ;EXPECTED STATUS
6854  025552  104346                          ERROR   346                     ;HCE W/HCI SET
6855  025554  000137  026214                  JMP     350$
6856  025560                          155$:
6857  025560  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6858  025564  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6859  025566  000240                          NOP                     ;RETURN HERE IF ERROR
6860  025570  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6861  025572  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6862  025574  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6863  025600                          160$:
6864
6865  025600                          170$:
6866  025600  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6867  025604  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
6868  025606  000240                          NOP                     ;RETURN HERE IF ERROR
6869  025610  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6870  025612  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6871  025614  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6872  025620                          180$:
6873
6874                                  ;CHANGE LOOP ADDRESSES
6875  025620  012737  025634  001122          MOV     #190$,$LPADR
6876  025626  012737  025634  001124          MOV     #190$,$LPERR
6877
6878                                  ;;*****************************************************************
6879                                  ;LOOP #3        READ
6880
6881  025634                          190$:
6882
6883                                  ;PREPARE DEVICE FOR READ OPERATION
6884  025634  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6885  025640  054130                          .WORD   054130  ;TASK DESCRIPTOR
6886  025642  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
6887  025644  000240                          NOP                     ;RETURN HERE IF ERROR
6888  025646  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6889  025650  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6890  025654                          200$:
6891
6892  025654                          240$:
6893
6894                                  ;READ DATA FROM DEVICE
```

```
6895  025654  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
6896  025662  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;LOAD STARTING BUFFER ADDRESS
6897  025670  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6898  025674  112722  000006                  MOVB    #RMDA,(R2)+
6899  025700  112722  000032                  MOVB    #RMOF,(R2)+
6900  025704  112722  000034                  MOVB    #RMDC,(R2)+
6901  025710  112722  000004                  MOVB    #RMBA,(R2)+
6902  025714  112722  000002                  MOVB    #RMWC,(R2)+
6903  025720  112722  000000                  MOVB    #RMCS1,(R2)+
6904  025724  112712  000200                  MOVB    #200,(R2)
6905
6906  025730  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6907  025734  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6908  025736  000240                          NOP                     ;RETURN HERE IF ERROR
6909  025740  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6910  025742  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6911  025746                          250$:
6912
6913                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6914  025746  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
6915
6916                                  ;WAIT FOR READ OPERATION TO COMPLETE
6917  025752  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
6918
6919                                  ;GO READ STATUS FOR READ OPERATION
6920  025756  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6921  025762  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6922  025764  000240                          NOP                     ;RETURN HERE IF ERROR
6923  025766  104000                          ERROP                   ;ERROR # DEFINED BY GET SUBROUTINE
6924  025770  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6925  025774                          260$:
6926
6927                                  ;CHECK FOR ERRORS DURING READ OPERATION
6928  025774  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6929  026000  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
6930  026002  000240                          NOP                     ;RETURN HERE IF ERROR
6931  026004  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6932  026006  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6933  026010  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6934  026014                          270$:
6935  026014  032737  000220  001344          BIT     #HCE!FER,RMER1I         ;ANY ERROR??
6936  026022  001407                          BEQ     271$                    ;NO!!
6937  026024  013737  001344  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6938  026032  042737  177557  001142          BIC     #^C<HCE!FER>,$BDDAT     ;CLEAR DONT CARES
6939  026040  000412                          BR      272$
6940  026042  032737  100000  001372  271$:   BIT     #BSE,RMER2I             ;ANY BAD SECTOR ERROR ??
6941  026050  001414                          BEQ     275$                    ;NO !!
6942  026052  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6943  026060  042737  077777  001142          BIC     #^CBSE,$BDDAT           ;CLEAR DONT CARES
6944  026066  012737  000000  001140  272$:   MOV     #0,$GDDAT               ;EXPECTED STATUS
6945  026074  104346                          ERROR   346                     ;HCE W/HCI SET
6946  026076  000137  026214                  JMP     350$
6947  026102                          275$:
6948  026102  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6949  026106  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
6950  026110  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
6951  026112  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6952  026114  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6953  026116  000137  026214          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6954  026122                  280$:
6955
6956  026122                  290$:
6957  026122  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6958  026126  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
6959  026130  000240                  NOP                     ;RETURN HERE IF ERROR
6960  026132  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6961  026134  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6962  026136  000137  026214          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6963  026142                  300$:
6964  026142  004737  043104          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
6965  026146  107552                  .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
6966  026150  110556                  .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
6967  026152  000404                  BR      310$                    ;GO TO 310$ IF NO ERROR
6968  026154  000240                  NOP                     ;RETURN HERE IF ERROR
6969  026156  104000                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
6970  026160  000137  026214          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6971  026164                  3'0$:
6972
6973  026164  006303                  ASL     R3                      ;SHIFT HCE BIT
6974  026166  001006                  BNE     320$                    ;CONTINUE IF NOT DONE
6975  026170  005704                  TST     R4                      ;SECOND HEADER DONE??
6976  026172  001011                  BNE     355$                    ;YES!!
6977  026174  012703  000001          MOV     #1,R3                   ;START WITH BIT 0
6978  026200  012704  000002          MOV     #2,R4                   ;DO SECOND HEADER WORD
6979  026204                  320$:
6980  026204  000137  024776          JMP     15$
6981  026210  000240                  NOP
6982  026212  000240                  NOP
6983  026214                  350$:
6984  026214  000471                  BR      370$            ;EXIT IF ERROR
6985  026216                  355$:
6986  026216  013737  001476  001434  MOV     ASNDC,RMDCO             ;
6987  026224  013737  001500  001406  MOV     ASNDA,RMDAO             ;
6988  026232  012737  107546  001404  MOV     #BUFONE,RMBAO           ;BUFFER ADDRESS
6989  026240  012737  177776  001402  MOV     #-2,RMWCO               ;WORD COUNT
6990  026246  012737  010000  001432  MOV     #FMT16,RMOFO            ;IN 16 BIT MODE
6991  026254  012737  000062  001400  MOV     #WH,RMCS10              ;FORMAT COMMAND
6992  026262  004737  042640          JSR     PC,GENBUF               ;SET UP BUFFER
6993  026266  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6994  026272  054130                  .WORD   054130  ;TASK DESCRIPTOR
6995  026274  000404                  BR      360$                    ;GO TO 360$ IF NO ERROR
6996  026276  000240                  NOP                     ;RETURN HERE IF ERROR
6997  026300  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6998  026302  000137  026306          JMP     360$                    ;GO TO 360$ IF ERROR WAS FOUND
6999  026306                  360$:
7000  026306  012737  000063  001400  MOV     #WH!GO,RMCS10           ;
7001  026314  012702  001535          MOV     #PUTINX,R2              ;TABLE ADDRESS
7002  026320  112722  000006          MOVB    #RMDA,(R2)+             ;
7003  026324  112722  000032          MOVB    #RMOF,(R2)+             ;
7004  026330  112722  000034          MOVB    #RMDC,(R2)+             ;
7005  026334  112722  000004          MOVB    #RMBA,(R2)+             ;
7006  026340  112722  000002          MOVB    #RMWC,(R2)+             ;
```

```
7007  026344  112722  000000          MOVB    #RMCS1,(R2)+            ;
7008  026350  112722  000200          MOVB    #200,(R2)+              ;
7009  026354  004737  044006          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7010  026360  000404                  BR      365$    ;GO TO 365$ IF NO ERROR
7011  026362  000240                  NOP                     ;RETURN HERE IF ERROR
7012  026364  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7013  026366  000137  026400          JMP     370$    ;GO TO 370$ IF ERROR WAS FOUND
7014  026372  000240          365$:   NOP
7015  026374  004737  044346          JSR     PC,TIMOUT               ;
7016  026400  012737  024766  001122  370$:   MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
7017  026406  012737  024766  001124          MOV     #10$,$LPERR
7018                                  ;;*****************************************************************
7019                                  ;*TEST 20        WRITE, READ W/ IVC ERROR
7020                                  ;;*****************************************************************
7021  026414                          TST20:
7022  026414  000004                  SCOPE                   ;SCOPE CALL
7023  026416  000240                  NOP                     ;START OF TEST
7024  026420  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7025  026424  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7026  026430  013701  001450          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7027  026434  012737  000020  001226  MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7028
7029                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7030  026442  012737  000000  001434  MOV     #0,RMDC0                ;CYLINDER = *
7031  026450  012737  000000  001406  MOV     #0,RMDA0                ;TRACK = *, SECTOR = *
7032  026456  012737  107546  001404  MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7033  026464  012737  177522  001402  MOV     #<^C256+1>,RMWC0        ;WORD COUNT
7034  026472  012737  010000  001432  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7035  026500  012737  000060  001400  MOV     #WD,RMCS1O              ;WRITE HEADER AND DATA COMMAND
7036
7037                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7038  026506  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7039
7040
7041                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7042  026512  012737  026530  001122  MOV     #70$,$LPADR
7043  026520  012737  026530  001124  MOV     #70$,$LPERR
7044  026526  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
7045
7046                                  ;;*****************************************************************
7047                                  ;LOOP #2        WRITE,READ
7048
7049  026530                          70$:
7050
7051
7052                                  ;PREPARE DEVICE FOR WRITE OPERATION
7053  026530  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7054  026534  054130                  .WORD   054130  ;TASK DESCRIPTOR
7055  026536  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
7056  026540  000240                  NOP                     ;RETURN HERE IF ERROR
7057  026542  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7058  026544  000137  027506          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7059  026550                          80$:
7060
7061                                  ;RESET VOLUME VALID
7062  026550  112737  000024  001535  MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
```

```
7063  026556  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7064  026564  012737  000001  001424          MOV     #DMD,RMMR10     ;RMMR1=DMD
7065  026572  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7066  026576  000404                          BR      90$             ;GO TO 90$ IF NO ERROR
7067  026600  000240                          NOP                     ;RETURN HERE IF ERROR
7068  026602  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7069  026604  000137  027506                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7070  026610                          90$:
7071  026610  112737  000024  001535          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7072  026616  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7073  026624  012737  000000  001424          MOV     #0,RMMR10       ;RMMR1=0
7074  026632  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7075  026636  000404                          BR      100$            ;GO TO 100$ IF NO ERROR
7076  026640  000240                          NOP                     ;RETURN HERE IF ERROR
7077  026642  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7078  026644  000137  027506                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7079  026650                          100$:
7080
7081  026650                          120$:
7082
7083                                  ;WRITE DATA TO THE DRIVE
7084  026650  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7085  026656  012737  107552  001404          MOV     #BUFONE+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
7086  026664  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7087  026670  112722  000006                  MOVB    #RMDA,(R2)+
7088  026674  112722  000034                  MOVB    #RMDC,(R2)+
7089  026700  112722  000032                  MOVB    #RMOF,(R2)+
7090  026704  112722  000004                  MOVB    #RMBA,(R2)+
7091  026710  112722  000002                  MOVB    #RMWC,(R2)+
7092  026714  112722  000000                  MOVB    #RMCS1,(R2)+
7093  026720  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
7094
7095  026724  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7096  026730  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
7097  026732  000240                          NOP                     ;RETURN HERE IF ERROR
7098  026734  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7099  026736  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7100  026742                          130$:
7101
7102                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7103  026742  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7104
7105                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7106  026746  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7107
7108                                  ;GO READ STATUS FOR WRITE COMMAND
7109  026752  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7110  026756  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
7111  026760  000240                          NOP                     ;RETURN HERE IF ERROR
7112  026762  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7113  026764  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7114  026770                          140$:
7115
7116                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7117  026770  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7118  026774  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
7119  026776  000240                          NOP                     ;RETURN HERE IF ERROR
7120  027000  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7121  027002  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7122  027004  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7123  027010                          150$:
7124  027010  032737  010000  001372          BIT     #IVC,RMER21             ;WAS "IVC" DETECTED??
7125  027016  001024                          BNE     170$                    ;YES!!
7126  027020  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7127  027024  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
7128  027026  000240                          NOP                     ;RETURN HERE IF ERROR
7129  027030  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7130  027032  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7131  027034  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7132  027040                          160$:
7133
7134  027040  013737  001372  001142          MOV     RMER21,$BDDAT           ;RECEIVED STATUS
7135  027046  013737  001372  001140          MOV     RMER21,$GDDAT           ;EXPECTED STATUS
7136  027054  052737  010000  001140          BIS     #IVC,$GDDAT
7137  027062  104342                          ERROR   342                     ;IVC NOT DETECTED
7138  027064  000137  027506                  JMP     350$
7139  027070                          170$:
7140  027070  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7141  027074  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
7142  027076  000240                          NOP                     ;RETURN HERE IF ERROR
7143  027100  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7144  027102  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7145  027104  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7146  027110                          180$:
7147
7148                                  ;CHANGE LOOP ADDRESSES
7149  027110  012737  027126  001122          MOV     #190$,$LPADR
7150  027116  012737  027126  001124          MOV     #190$,$LPERR
7151  027124  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7152
7153                                  ;;*****************************************************************
7154                                  ;LOOP #3        READ
7155
7156  027126                          190$:
7157
7158                                  ;PREPARE DEVICE FOR READ OPERATION
7159  027126  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7160  027132  054130                          .WORD   054130  ;TASK DESCRIPTOR
7161  027134  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7162  027136  000240                          NOP                     ;RETURN HERE IF ERROR
7163  027140  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7164  027142  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7165  027146                          200$:
7166
7167                                  ;RESET VOLUME VALID
7168  027146  112737  000024  001535          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7169  027154  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7170  027162  012737  000001  001424          MOV     #DMD,RMMR10     ;RMMR1=DMD
7171  027170  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7172  027174  000404                          BR      210$            ;GO TO 210$ IF NO ERROR
7173  027176  000240                          NOP                     ;RETURN HERE IF ERROR
7174  027200  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
```

```
7175  027202  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7176  027206                            210$:
7177  027206  112737  000024  001535            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7178  027214  112737  000200  001536            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7179  027222  012737  000000  001424            MOV     #0,RMMR10       ;RMMR1=0
7180  027230  004737  044006                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7181  027234  000404                            BR      220$            ;GO TO 220$ IF NO ERROR
7182  027236  000240                            NOP                     ;RETURN HERE IF ERROR
7183  027240  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7184  027242  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7185  027246                            220$:
7186  027246                            240$:
7187
7188                                    ;READ DATA FROM DEVICE
7189  027246  012737  000071  001400            MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
7190  027254  012737  110556  001404            MOV     #BUFTWO+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
7191  027262  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7192  027266  112722  000006                    MOVB    #RMDA,(R2)+
7193  027272  112722  000032                    MOVB    #RMOF,(R2)+
7194  027276  112722  000034                    MOVB    #RMDC,(R2)+
7195  027302  112722  000004                    MOVB    #RMBA,(R2)+
7196  027306  112722  000002                    MOVB    #RMWC,(R2)+
7197  027312  112722  000000                    MOVB    #RMCS1,(R2)+
7198  027316  112712  000200                    MOVB    #200,(R2)
7199
7200  027322  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7201  027326  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
7202  027330  000240                            NOP                     ;RETURN HERE IF ERROR
7203  027332  104000                            ERROP                   ;ERROR # DEFINED BY PUT SUBROUTINE
7204  027334  000137  027506                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7205  027340                            250$:
7206
7207                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7208  027340  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7209
7210                                    ;WAIT FOR READ OPERATION TO COMPLETE
7211  027344  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7212
7213                                    ;GO READ STATUS FOR READ OPERATION
7214  027350  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7215  027354  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
7216  027356  000240                            NOP                     ;RETURN HERE IF ERROR
7217  027360  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7218  027362  000137  027506                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7219  027366                            260$:
7220
7221                                    ;CHECK FOR ERRORS DURING READ OPERATION
7222  027366  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7223  027372  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
7224  027374  000240                            NOP                     ;RETURN HERE IF ERROR
7225  027376  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7226  027400  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7227  027402  000137  027506                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7228  027406                            270$:
7229  027406  032737  010000  001372            BIT     #IVC,RMER21             ;WAS "IVC" DETECTED??
7230  027414  001024                            BNE     290$                    ;YES!!
```

```
7231  027416  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7232  027422  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
7233  027424  000240                  NOP                     ;RETURN HERE IF ERROR
7234  027426  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7235  027430  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7236  027432  000137  027506          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7237  027436                  280$:
7238  027436  013737  001372  001140  MOV     RMER21,$GDDAT           ;EXPECTED STATUS
7239  027444  052737  010000  001140  BIS     #IVC,$GDDAT
7240  027452  013737  001372  001142  MOV     RMER21,$BDDAT           ;RECEIVED STATUS
7241  027460  104342                  ERROR   342                     ;IVC NOT DETECTED
7242  027462  000137  027506          JMP     350$
7243
7244  027466                  290$:
7245  027466  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7246  027472  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
7247  027474  000240                  NOP                     ;RETURN HERE IF ERROR
7248  027476  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7249  027500  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7250  027502  000137  027506          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7251  027506                  300$:
7252
7253  027506                  350$:
7254  027506  012737  026530  001122  MOV     #70$,$LPADR             ;CHANGE LOOP TO START OF TEST
7255  027514  012737  026530  001124  MOV     #70$,$LPERR
7256                          ;;*****************************************************************
7257                          ;*TEST 21        WRITE, READ W/ ABORT
7258                          ;;*****************************************************************
7259  027522                  TST21:
7260  027522  000004                  SCOPE                   ;SCOPE CALL
7261  027524  000240                  NOP                     ;START OF TEST
7262  027526  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7263  027532  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7264  027536  013701  001450          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7265  027542  012737  000021  001226  MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7266
7267
7268                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7269  027550  012737  000000  001434  MOV     #0,RMDC0                ;CYLINDER = 0
7270  027556  012737  000000  001406  MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
7271  027564  012737  107546  001404  MOV     #BUF0NE,RMBA0           ;BUS ADDRESS
7272  027572  012737  177400  001402  MOV     #<^C256.+1>,RMWC0       ;WORD COUNT
7273  027600  012737  010000  001432  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7274  027606  012737  000063  001400  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
7275
7276                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7277  027614  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7278
7279                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7280  027620  012737  027636  001122  MOV     #70$,$LPADR
7281  027626  012737  027636  001124  MOV     #70$,$LPERR
7282  027634  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
7283
7284                          ;;*****************************************************************
7285                          ;LOOP #2        WRITE,READ
7286
```

```
7287  027636                          70$:
7288
7289
7290                                  ;PREPARE DEVICE FOR WRITE OPERATION
7291  027636  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7292  027642  054130                        .WORD   054130  ;TASK DESCRIPTOR
7293  027644  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
7294  027646  000240                        NOP                     ;RETURN HERE IF ERROR
7295  027650  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7296  027652  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7297  027656                          80$:
7298
7299                                  ;SET UNSAFE ERROR
7300  027656  112737  000014  001535        MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
7301  027664  112737  000200  001536        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7302  027672  012737  040000  001414        MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER - UNS
7303  027700  004737  044006                JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
7304  027704  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7305  027706  000240                        NOP                     ;RETURN HERE TO REPORT ERROR
7306  027710  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7307  027712  000137  030470                JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
7308  027716                          90$:
7309
7310  027716                          120$:
7311
7312                                  ;WRITE DATA TO THE DRIVE
7313  027716  012737  000061  001400        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7314  027724  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7315  027730  112722  000006                MOVB    #RMDA,(R2)+
7316  027734  112722  000034                MOVB    #RMDC,(R2)+
7317  027740  112722  000032                MOVB    #RMOF,(R2)+
7318  027744  112722  000004                MOVB    #RMBA,(R2)+
7319  027750  112722  000002                MOVB    #RMWC,(R2)+
7320  027754  112722  000000                MOVB    #RMCS1,(R2)+
7321  027760  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
7322
7323
7324                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7325  027764  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7326  027770  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7327  027774  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
7328  027776  000240                        NOP                     ;RETURN HERE IF ERROR
7329  030000  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7330  030002  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7331  030006                          130$:
7332  030006  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7333  030012  000404                        BR      135$                    ;GO TO 135$ IF NO ERROR
7334  030014  000240                        NOP                     ;RETURN HERE IF ERROR
7335  030016  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7336  030020  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7337  030024  032737  020000  001342  135$:   BIT     #PIP,RMDS1              ;WAS COMMAND EXECUTED?
7338  030032  001412                        BEQ     136$
7339  030034  013737  001342  001140        MOV     RMDS1,$GDDAT            ;EXPECTED STATUS
7340  030042  042737  020000  001140        BIC     #PIP,$GDDAT
7341  030050  013737  001342  001142        MOV     RMDS1,$BDDAT            ;RECEIVED STATUS
7342  030056  104347                        ERROR   347                     ;NO ABORT
```

```
7343  030060                          136$:
7344
7345                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7346  030060  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7347
7348                                  ;GO READ STATUS FOR WRITE COMMAND
7349  030064  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7350  030070  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
7351  030072  000240                        NOP                     ;RETURN HERE IF ERROR
7352  030074  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7353  030076  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7354  030102                          140$:
7355
7356                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7357  030102  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7358  030106  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7359  030110  000240                        NOP                     ;RETURN HERE IF ERROR
7360  030112  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7361  030114  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7362  030116  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7363  030122                          150$:
7364
7365  030122  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7366  030126  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
7367  030130  000240                        NOP                     ;RETURN HERE IF ERROR
7368  030132  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7369  030134  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7370  030136  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7371  030142                          180$:
7372
7373                                  ;CHANGE LOOP ADDRESSES
7374  030142  012737  030160  001122        MOV     #190$,$LPADR
7375  030150  012737  030160  001124        MOV     #190$,$LPERR
7376  030156  000410                        BR      200$                    ;SKIP TO NEXT OPERATION
7377
7378                                  ;;*****************************************************************
7379                                  ;LOOP #3        READ
7380
7381  030160                          190$:
7382
7383                                  ;PREPARE DEVICE FOR READ OPERATION
7384  030160  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7385  030164  054130                        .WORD   054130  ;TASK DESCRIPTOR
7386  030166  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
7387  030170  000240                        NOP                     ;RETURN HERE IF ERROR
7388  030172  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7389  030174  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7390  030200                          200$:
7391                                  ;SET UNSAFE ERROR
7392  030200  112737  000014  001535        MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
7393  030206  112737  000200  001536        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7394  030214  012737  040000  001414        MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
7395  030222  004737  044006                JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
7396  030226  000404                        BR      210$    ;GO TO 210$ IF NO ERROR
7397  030230  000240                        NOP                     ;RETURN HERE TO REPORT ERROR
7398  030232  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
```

```
 7399  030234  000137  030470                    JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
 7400  030240                                210$:
 7401
 7402  030240                                240$:
 7403
 7404                                        ;READ DATA FROM DEVICE
 7405  030240  012737  000071  001400            MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
 7406  030246  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
 7407  030252  112722  000006                    MOVB    #RMDA,(R2)+
 7408  030256  112722  000032                    MOVB    #RMOF,(R2)+
 7409  030262  112722  000034                    MOVB    #RMDC,(R2)+
 7410  030266  112722  000004                    MOVB    #RMBA,(R2)+
 7411  030272  112722  000002                    MOVB    #RMWC,(R2)+
 7412  030276  112722  000000                    MOVB    #RMCS1,(R2)+
 7413  030302  112712  000200                    MOVB    #200,(R2)
 7414
 7415                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 7416  030306  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 7417
 7418  030312  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 7419  030316  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
 7420  030320  000240                            NOP                     ;RETURN HERE IF ERROR
 7421  030322  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 7422  030324  000137  030470                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 7423  030330                                250$:
 7424
 7425                                        ;SEE IF DEVICE STARTED COMMAND
 7426  030330  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 7427  030334  000404                            BR      255$                    ;GO TO 255$ IF NO ERROR
 7428  030336  000240                            NOP                     ;RETURN HERE IF ERROR
 7429  030340  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 7430  030342  000137  030470                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 7431  030346  032737  020000  001342  255$:     BIT     #PIP,RMDSI              ;WAS COMMAND EXECUTED??
 7432  030354  001414                            BEQ     256$                    ;NO!!
 7433  030356  013737  001342  001140            MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
 7434  030364  042737  020000  001140            BIC     #PIP,$GDDAT
 7435  030372  013737  001342  001142            MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
 7436  030400  104347                            ERROR   347                     ;NO ABORT
 7437  030402  000137  030470                   ..JMP     350$
 7438  030406                                256$:
 7439
 7440                                        ;WAIT FOR READ OPERATION TO COMPLETE
 7441  030406  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 7442
 7443                                        ;GO READ STATUS FOR READ OPERATION
 7444  030412  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 7445  030416  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
 7446  030420  000240                            NOP                     ;RETURN HERE IF ERROR
 7447  030422  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 7448  030424  000137  030470                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 7449  030430                                260$:
 7450
 7451                                        ;CHECK FOR ERRORS DURING READ OPERATION
 7452  030430  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7453  030434  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
 7454  030436  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
7455  030440  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7456  030442  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7457  030444  000137  030470                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7458  030450                          270$:
7459  030450  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7460  030454  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
7461  030456  000240                          NOP                     ;RETURN HERE IF ERROR
7462  030460  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7463  030462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7464  030464  000137  030470                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7465  030470                          300$:
7466
7467  030470                          350$:
7468                                  ;;*******************************************************************
7469                                  ;*TEST 22        WRITE, READ W/ BAI
7470                                  ;;*******************************************************************
7471  030470                          TST22:
7472  030470  000004                          SCOPE                   ;SCOPE CALL
7473  030472  000240                          NOP                     ;START OF TEST
7474  030474  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7475  030500  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7476  030504  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7477  030510  012737  000022  001226          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7478
7479                                  ;;*******************************************************************
7480                                  ;LOOP #1        FORMAT,WRITE,READ
7481
7482  030516                          10$:
7483
7484                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
7485  030516  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7486  030522  054130                          .WORD   054130  ;TASK DESCRIPTOR
7487  030524  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
7488  030526  000240                          NOP                     ;RETURN HERE IF ERROR
7489  030530  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7490  030532  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7491  030536                          20$:
7492
7493                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7494  030536  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
7495  030544  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
7496  030552  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7497  030560  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
7498  030566  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7499  030574  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
7500
7501                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
7502
7503  030602  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
7504  030606  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
7505  030610  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
7506  030614  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
7507  030616  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7508  030622                          25$:
7509  030622  012737  071044  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
7510  030630  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
```

```
7511  030636  004737  042640              JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
7512
7513                                      ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
7514  030642  012702  001535              MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
7515  030646  112722  000034              MOVB    #RMDC,(R2)+
7516  030652  112722  000006              MOVB    #RMDA,(R2)+
7517  030656  112722  000004              MOVB    #RMBA,(R2)+
7518  030662  112722  000002              MOVB    #RMWC,(R2)+
7519  030666  112722  000032              MOVB    #RMOF,(R2)+
7520  030672  112722  000000              MOVB    #RMCS1,(R2)+
7521  030676  112712  000200              MOVB    #200,(R2)               ;TERMINATE TABLE
7522  030702                              30$:
7523
7524                                      ;FORMAT THE DRIVE
7525  030702  004737  044006              JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7526  030706  000404                      BR      40$                     ;GO TO 40$ IF NO ERROR
7527  030710  000240                      NOP                     ;RETURN HERE IF ERROR
7528  030712  104000                      ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7529  030714  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7530  030720                              40$:
7531
7532                                      ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7533  030720  004737  043452              JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7534
7535                                      ;WAIT FOR THE FORMAT TO COMPLETE
7536  030724  004737  044346              JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
7537
7538                                      ;GO READ STATUS FOR FORMAT OPERATION
7539  030730  004737  043536              JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7540  030734  000404                      BR      50$                     ;GO TO 50$ IF NO ERROR
7541  030736  000240                      NOP                     ;RETURN HERE IF ERROR
7542  030740  104000                      ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7543  030742  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7544  030746                              50$:
7545
7546                                      ;VERIFY NO ERRORS DURING FORMAT
7547  030746  004737  057036              JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
7548  030752  000405                      BR      60$                     ;GO TO 60$ IF NO ERROR
7549  030754  000240                      NOP                     ;RETURN HERE IF ERROR
7550  030756  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7551  030760  004736                      JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7552  030762  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7553  030766                              60$:
7554
7555                                      ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7556  030766  012737  031004  001122      MOV     #70$,$LPADR
7557  030774  012737  031004  001124      MOV     #70$,$LPERR
7558  031002  000410                      BR      80$                     ;SKIP TO WRITE OPERATION
7559
7560                                      ;;*****************************************************************
7561                                      ;LOOP #2        WRITE,READ
7562
7563  031004                              70$:
7564
7565
7566                                      ;PREPARE DEVICE FOR WRITE OPERATION
```

```
7567  031004  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7568  031010  054130                  .WORD   054130  ;TASK DESCRIPTOR
7569  031012  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
7570  031014  000240                  NOP                     ;RETURN HERE IF ERROR
7571  031016  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7572  031020  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7573  031024                  80$:
7574
7575
7576  031024                  120$:
7577
7578                          ;WRITE DATA TO THE DRIVE
7579  031024  111102                  MOVB    (R1),R2                 ;GENERATE CS2
7580  031026  042702  177770          BIC     #^C<U0!U1!U2>,R2
7581  031032  042702  000010          BIC     #BAI,R2
7582  031036  010237  001410          MOV     R2,RMCS20               ;INHIBIT ADDRESS INCREMENT
7583  031042  012737  177400  001402  MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
7584  031050  012737  107552  001404  MOV     #BUFONE+4,RMBA0         ;CHANGE ADDRESS
7585  031056  012737  000061  001400  MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7586  031064  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7587  031070  112722  000010          MOVB    #RMCS2,(R2)+
7588  031074  112722  000006          MOVB    #RMDA,(R2)+
7589  031100  112722  000034          MOVB    #RMDC,(R2)+
7590  031104  112722  000032          MOVB    #RMOF,(R2)+
7591  031110  112722  000004          MOVB    #RMBA,(R2)+
7592  031114  112722  000002          MOVB    #RMWC,(R2)+
7593  031120  112722  000000          MOVB    #RMCS1,(R2)+
7594  031124  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
7595
7596  031130  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7597  031134  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
7598  031136  000240                  NOP                     ;RETURN HERE IF ERROR
7599  031140  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7600  031142  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7601  031146                  130$:
7602
7603                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7604  031146  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7605
7606                          ;WAIT FOR WRITE COMMAND TO COMPLETE
7607  031152  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7608
7609                          ;GO READ STATUS FOR WRITE COMMAND
7610  031156  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7611  031162  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
7612  031164  000240                  NOP                     ;RETURN HERE IF ERROR
7613  031166  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7614  031170  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7615  031174                  140$:
7616
7617                          ;CHECK FOR ERRORS DURING WRITE OPERATION
7618  031174  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7619  031200  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
7620  031202  000240                  NOP                     ;RETURN HERE IF ERROR
7621  031204  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7622  031206  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
7623  031210  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7624  031214                          150$:
7625  031214  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7626  031220  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
7627  031222  000240                          NOP                     ;RETURN HERE IF ERROR
7628  031224  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7629  031226  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7630  031230  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7631  031234                          160$:
7632
7633  031234                          170$:
7634  031234  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7635  031240  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
7636  031242  000240                          NOP                     ;RETURN HERE IF ERROR
7637  031244  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7638  031246  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7639  031250  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7640  031254                          180$:
7641
7642                                  ;CHANGE LOOP ADDRESSES
7643  031254  012737  031272  001122          MOV     #190$,$LPADR
7644  031262  012737  031272  001124          MOV     #190$,$LPERR
7645  031270  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7646
7647                                  ;;*****************************************************************
7648                                  ;LOOP #3        READ
7649
7650  031272                          190$:
7651
7652                                  ;PREPARE DEVICE FOR READ OPERATION
7653  031272  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7654  031276  054130                          .WORD   054130  ;TASK DESCRIPTOR
7655  031300  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7656  031302  000240                          NOP                     ;RETURN HERE IF ERROR
7657  031304  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7658  031306  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7659  031312                          200$:
7660
7661  031312                          240$:
7662
7663                                  ;READ DATA FROM DEVICE
7664  031312  111102                          MOVB    (R1),R2                 ;RESET BAI
7665  031314  042702  177770                  BIC     #^C<U0!U1!U2>,R2
7666  031320  010237  001410                  MOV     R2,RMCS20
7667  031324  012737  110556  001404          MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
7668  031332  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
7669  031340  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7670  031344  112722  000006                  MOVB    #RMDA,(R2)+
7671  031350  112722  000032                  MOVB    #RMOF,(R2)+
7672  031354  112722  000034                  MOVB    #RMDC,(R2)+
7673  031360  112722  000004                  MOVB    #RMBA,(R2)+
7674  031364  112722  000002                  MOVB    #RMWC,(R2)+
7675  031370  112722  000000                  MOVB    #RMCS1,(R2)+
7676  031374  112712  000200                  MOVB    #200,(R2)
7677
7678  031400  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
7679  031404  000404                    BR      250$                    ;GO TO 250$ IF NO ERROR
7680  031406  000240                    NOP                     ;RETURN HERE IF ERROR
7681  031410  104000                    ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7682  031412  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7683  031416                    250$:
7684
7685                            ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7686  031416  004737  043452            JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7687
7688                            ;WAIT FOR READ OPERATION TO COMPLETE
7689  031422  004737  044346            JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7690
7691                            ;GO READ STATUS FOR READ OPERATION
7692  031426  004737  043536            JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7693  031432  000404                    BR      260$                    ;GO TO 260$ IF NO ERROR
7694  031434  000240                    NOP                     ;RETURN HERE IF ERROR
7695  031436  104000                    ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7696  031440  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7697  031444                    260$:
7698
7699                            ;CHECK FOR ERRORS DURING READ OPERATION
7700  031444  004737  044532            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7701  031450  000405                    BR      270$                    ;GO TO 270$ IF NO ERROR
7702  031452  000240                    NOP                     ;RETURN HERE IF ERROR
7703  031454  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7704  031456  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7705  031460  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7706  031464                    270$:
7707  031464  004737  057036            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7708  031470  000405                    BR      280$                    ;GO TO 280$ IF NO ERROR
7709  031472  000240                    NOP                     ;RETURN HERE IF ERROR
7710  031474  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7711  031476  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7712  031500  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7713  031504                    280$:
7714
7715  031504                    290$:
7716  031504  004737  045364            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7717  031510  000405                    BR      300$                    ;GO TO 300$ IF NO ERROR
7718  031512  000240                    NOP                     ;RETURN HERE IF ERROR
7719  031514  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7720  031516  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7721  031520  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7722  031524                    300$:
7723  031524  004737  043104            JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
7724  031530  107552                    .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
7725  031532  110556                    .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
7726  031534  000404                    BR      310$                    ;GO TO 310$ IF NO ERROR
7727  031536  000240                    NOP                     ;RETURN HERE IF ERROR
7728  031540  104000                    ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
7729  031542  000137  031546            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7730  031546                    310$:
7731
7732  031546                    350$:
7733                            ;;*****************************************************************
7734                            ;*TEST 23        WRITE, READ EACH CURRENT LEVEL
```

```
7735                                    ;;*****************************************************************
7736  031546                            TST23:
7737  031546  000004                            SCOPE                   ;SCOPE CALL
7738  031550  000240                            NOP                     ;START OF TEST
7739  031552  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7740  031556  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7741  031562  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7742  031566  012737  000023  001226            MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7743
7744                                    ;;*****************************************************************
7745                                    ;LOOP #1        FORMAT,WRITE,READ
7746
7747  031574                            10$:
7748
7749                                    ;PREPARE THE DEVICE FOR FORMAT OPERATION
7750  031574  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7751  031600  054130                            .WORD   054130  ;TASK DESCRIPTOR
7752  031602  000404                            BR      20$                     ;GO TO 20$ IF NO ERROR
7753  031604  000240                            NOP                     ;RETURN HERE IF ERROR
7754  031606  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7755  031610  000137  032620                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7756  031614                            20$:
7757
7758                                    ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7759  031614  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
7760  031622  012737  000000  001406    25$:    MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
7761  031630  012737  107546  001404            MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7762  031636  012737  177376  001402            MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
7763  031644  012737  010000  001432            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7764  031652  012737  000063  001400            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
7765
7766                                    ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
7767
7768  031660  004737  040734                    JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
7769  031664  000405                            BR      26$                     ;GO TO 26$ IF NO ERROR
7770  031666  104401  067464                    TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
7771  031672  104000                            ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
7772  031674  000137  032620                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7773  031700                            26$:
7774  031700  012737  071106  001174            MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
7775  031706  012737  000001  001176            MOV     #1,$TMP1                ;RANGE OF PATTERN
7776  031714  004737  042640                    JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
7777
7778                                    ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
7779  031720  012702  001535                    MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
7780  031724  112722  000034                    MOVB    #RMDC,(R2)+
7781  031730  112722  000006                    MOVB    #RMDA,(R2)+
7782  031734  112722  000004                    MOVB    #RMBA,(R2)+
7783  031740  112722  000002                    MOVB    #RMWC,(R2)+
7784  031744  112722  000032                    MOVB    #RMOF,(R2)+
7785  031750  112722  000000                    MOVB    #RMCS1,(R2)+
7786  031754  112712  000200                    MOVB    #200,(R2)               ;TERMINATE TABLE
7787  031760                            30$:
7788
7789                                    ;FORMAT THE DRIVE
7790  031760  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
7791  031764  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
7792  031766  000240                        NOP                     ;RETURN HERE IF ERROR
7793  031770  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7794  031772  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7795  031776                        40$:
7796
7797                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7798  031776  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7799
7800                                ;WAIT FOR THE FORMAT TO COMPLETE
7801  032002  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7802
7803                                ;GO READ STATUS FOR FORMAT OPERATION
7804  032006  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7805  032012  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
7806  032014  000240                        NOP                     ;RETURN HERE IF ERROR
7807  032016  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7808  032020  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7809  032024                        50$:
7810
7811                                ;VERIFY NO ERRORS DURING FORMAT
7812  032024  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7813  032030  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
7814  032032  000240                        NOP                     ;RETURN HERE IF ERROR
7815  032034  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7816  032036  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7817  032040  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7818  032044                        60$:
7819
7820                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7821  032044  012737  032062  001122        MOV     #70$,$LPADR
7822  032052  012737  032062  001124        MOV     #70$,$LPERR
7823  032060  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
7824
7825                                ;;*****************************************************************
7826                                ;LOOP #2        WRITE,READ
7827
7828  032062                        70$:
7829
7830
7831                                ;PREPARE DEVICE FOR WRITE OPERATION
7832  032062  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7833  032066  054130                        .WORD   054130  ;TASK DESCRIPTOR
7834  032070  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
7835  032072  000240                        NOP                     ;RETURN HERE IF ERROR
7836  032074  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7837  032076  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7838  032102                        80$:
7839
7840
7841  032102                        120$:
7842
7843                                ;WRITE DATA TO THE DRIVE
7844  032102  012737  177400  001402        MOV     #<^C256.*1>,RMWCO       ;CHANGE WORD COUNT
7845  032110  012737  107552  001404        MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
7846  032116  012737  000061  001400        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
```

```
7847  032124  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7848  032130  112722  000006          MOVB    #RMDA,(R2)+
7849  032134  112722  000034          MOVB    #RMDC,(R2)+
7850  032140  112722  000032          MOVB    #RMOF,(R2)+
7851  032144  112722  000004          MOVB    #RMBA,(R2)+
7852  032150  112722  000002          MOVB    #RMWC,(R2)+
7853  032154  112722  000000          MOVB    #RMCS1,(R2)+
7854  032160  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
7855
7856  032164  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7857  032170  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
7858  032172  000240                  NOP                     ;RETURN HERE IF ERROR
7859  032174  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7860  032176  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7861  032202                  130$:
7862
7863                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7864  032202  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7865
7866                          ;WAIT FOR WRITE COMMAND TO COMPLETE
7867  032206  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7868
7869                          ;GO READ STATUS FOR WRITE COMMAND
7870  032212  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7871  032216  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
7872  032220  000240                  NOP                     ;RETURN HERE IF ERROR
7873  032222  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7874  032224  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7875  032230                  140$:
7876
7877                          ;CHECK FOR ERRORS DURING WRITE OPERATION
7878  032230  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7879  032234  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
7880  032236  000240                  NOP                     ;RETURN HERE IF ERROR
7881  032240  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7882  032242  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7883  032244  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7884  032250                  150$:
7885  032250  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7886  032254  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
7887  032256  000240                  NOP                     ;RETURN HERE IF ERROR
7888  032260  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7889  032262  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7890  032264  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7891  032270                  160$:
7892
7893  032270                  170$:
7894  032270  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7895  032274  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
7896  032276  000240                  NOP                     ;RETURN HERE IF ERROR
7897  032300  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7898  032302  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7899  032304  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7900  032310                  180$:
7901
7902                          ;CHANGE LOOP ADDRESSES
```

```
7903  032310  012737  032326  001122          MOV     #190$,$LPADR
7904  032316  012737  032326  001124          MOV     #190$,$LPERR
7905  032324  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7906
7907                                  ;;*****************************************************************
7908                                  ;LOOP #3        READ
7909
7910  032326                          190$:
7911
7912                                  ;PREPARE DEVICE FOR READ OPERATION
7913  032326  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7914  032332  054130                          .WORD   054130  ;TASK DESCRIPTOR
7915  032334  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7916  032336  000240                          NOP                     ;RETURN HERE IF ERROR
7917  032340  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7918  032342  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7919  032346                          200$:
7920
7921  032346                          240$:
7922
7923                                  ;READ DATA FROM DEVICE
7924  032346  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;CHANGE ADDRESS
7925  032354  012737  177400  001402          MOV     #<^C256.+1>,RMWCO
7926  032362  012737  000071  001400          MOV     #RD!GO,RMCS1C           ;READ DATA COMMAND
7927  032370  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7928  032374  112722  000006                  MOVB    #RMDA,(R2)+
7929  032400  112722  000032                  MOVB    #RMOF,(R2)+
7930  032404  112722  000034                  MOVB    #RMDC,(R2)+
7931  032410  112722  000004                  MOVB    #RMBA,(R2)+
7932  032414  112722  000002                  MOVB    #RMWC,(R2)+
7933  032420  112722  000000                  MOVB    #RMCS1,(R2)+
7934  032424  112712  000200                  MOVB    #200,(R2)
7935
7936  032430  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7937  032434  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
7938  032436  000240                          NOP                     ;RETURN HERE IF ERROR
7939  032440  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7940  032442  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7941  032446                          250$:
7942
7943                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7944  032446  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7945
7946                                  ;WAIT FOR READ OPERATION TO COMPLETE
7947  032452  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7948
7949                                  ;GO READ STATUS FOR READ OPERATION
7950  032456  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7951  032462  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
7952  032464  000240                          NOP                     ;RETURN HERE IF ERROR
7953  032466  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7954  032470  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7955  032474                          260$:
7956
7957                                  ;CHECK FOR ERRORS DURING READ OPERATION
7958  032474  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
```

```
7959  032500  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
7960  032502  000240                        NOP                     ;RETURN HERE IF ERROR
7961  032504  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7962  032506  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7963  032510  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7964  032514                        270$:
7965  032514  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7966  032520  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
7967  032522  000240                        NOP                     ;RETURN HERE IF ERROR
7968  032524  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7969  032526  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7970  032530  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7971  032534                        280$:
7972
7973  032534                        290$:
7974  032534  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7975  032540  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
7976  032542  000240                        NOP                     ;RETURN HERE IF ERROR
7977  032544  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7978  032546  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7979  032550  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7980  032554                        300$:
7981  032554  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
7982  032560  107552                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
7983  032562  110556                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
7984  032564  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
7985  032566  000240                        NOP                     ;RETURN HERE IF ERROR
7986  032570  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
7987  032572  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7988  032576                        310$:
7989  032576  062737  000200  001434        ADD     #128.,RMDCO             ;ADVANCE TO NEXT THRESHOLD
7990  032604  023727  001434  001400        CMP     RMDCO,#768.             ;DONE??
7991  032612  101002                        BHI     350$                    ;YES!!
7992  032614  000137  031622                JMP     25$                     ;DO NEXT CURRENT LEVEL
7993
7994  032620                        350$:
7995  032620  012737  031574  001122        MOV     #10$,$LPADR             ;LOOP BACK TO START OF TEST
7996  032626  012737  031574  001124        MOV     #10$,$LPERR
7997                                ;;*****************************************************************
7998                                ;*TEST 24        WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING
7999                                ;;*****************************************************************
8000  032634                        TST24:
8001  032634  000004                        SCOPE                   ;SCOPE CALL
8002  032636  000240                        NOP                     ;START OF TEST
8003  032640  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8004  032644  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8005  032650  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8006  032654  012737  000024  001226        MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8007
8008                                ;;*****************************************************************
8009                                ;LOOP #1        FORMAT,WRITE,READ
8010
8011  032662                        10$:
8012
8013                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
8014  032662  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
8015  032666  054130                        .WORD   054130  ;TASK DESCRIPTOR
8016  032670  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
8017  032672  000240                        NOP                     ;RETURN HERE IF ERROR
8018  032674  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8019  032676  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8020  032702                        20$:
8021
8022                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8023  032702  012737  000577  001434        MOV     #383.,RMDC0             ;CYLINDER = 383.
8024  032710  012737  002037  001406        MOV     #2037,RMDA0             ;TRACK = 4, SECTOR = 31
8025  032716  012737  107546  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8026  032724  012737  176774  001402        MOV     #<^C<2**256.+2>>+1>,RMWC0               ;WORD COUNT
8027  032732  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8028  032740  012737  000063  001400        MOV     #WHIGO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8029
8030                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8031
8032  032746  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8033  032752  000405                        BR      25$                     ;GO TO 25$ IF NO ERROR
8034  032754  104401  067464                TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8035  032760  104000                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8036  032762  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8037  032766                        25$:
8038  032766  012737  071044  001174        MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
8039  032774  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
8040  033002  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8041
8042                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATION
8043  033006  012702  001535                MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8044  033012  112722  000034                MOVB    #RMDC,(R2)+
8045  033016  112722  000006                MOVB    #RMDA,(R2)+
8046  033022  112722  000004                MOVB    #RMBA,(R2)+
8047  033026  112722  000002                MOVB    #RMWC,(R2)+
8048  033032  112722  000032                MOVB    #RMOF,(R2)+
8049  033036  112722  000000                MOVB    #RMCS1,(R2)+
8050  033042  112712  000200                MOVB    #200,(R2)               ;TERMINATE TABLE
8051  033046                        30$:
8052
8053                                ;FORMAT THE DRIVE
8054  033046  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8055  033052  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
8056  033054  000240                        NOP                     ;RETURN HERE IF ERROR
8057  033056  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8058  033060  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8059  033064                        40$:
8060
8061                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8062  033064  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8063
8064                                ;WAIT FOR THE FORMAT TO COMPLETE
8065  033070  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8066
8067                                ;GO READ STATUS FOR FORMAT OPERATION
8068  033074  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8069  033100  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
8070  033102  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
8071  033104  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8072  033106  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8073  033112                          50$:
8074
8075                                  ;VERIFY NO ERRORS DURING FORMAT
8076  033112  004737  057034                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8077  033116  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
8078  033120  000240                        NOP                     ;RETURN HERE IF ERROR
8079  033122  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8080  033124  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8081  033126  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8082  033132                          60$:
8083
8084                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8085  033132  012737  033150  001122        MOV     #70$,$LPADR
8086  033140  012737  033150  001124        MOV     #70$,$LPERR
8087  033146  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
8088
8089                                  ;;*****************************************************************
8090                                  ;LOOP #2        WRITE,READ
8091
8092  033150                          70$:
8093
8094
8095                                  ;PREPARE DEVICE FOR WRITE OPERATION
8096  033150  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8097  033154  054130                        .WORD   054130  ;TASK DESCRIPTOR
8098  033156  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
8099  033160  000240                        NOP                     ;RETURN HERE IF ERROR
8100  033162  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8101  033164  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8102  033170                          80$:
8103
8104
8105  033170                          120$:
8106
8107                                  ;WRITE DATA TO THE DRIVE
8108  033170  012737  177000  001402        MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
8109  033176  012737  107552  001404        MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
8110  033204  012737  000061  001400        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
8111  033212  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8112  033216  112722  000006                MOVB    #RMDA,(R2)+
8113  033222  112722  000034                MOVB    #RMDC,(R2)+
8114  033226  112722  000032                MOVB    #RMOF,(R2)+
8115  033232  112722  000004                MOVB    #RMBA,(R2)+
8116  033236  112722  000002                MOVB    #RMWC,(R2)+
8117  033242  112722  000000                MOVB    #RMCS1,(R2)+
8118  033246  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
8119
8120  033252  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8121  033256  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
8122  033260  000240                        NOP                     ;RETURN HERE IF ERROR
8123  033262  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8124  033264  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8125  033270                          130$:
8126
```

```
                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
033270  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
033274  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
033300  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
033304  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
033306  000240                        NOP                     ;RETURN HERE IF ERROR
033310  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
033312  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033316                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
033316  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
033322  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
033324  000240                        NOP                     ;RETURN HERE IF ERROR
033326  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
033330  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033332  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033336                          150$:
033336  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
033342  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
033344  000240                        NOP                     ;RETURN HERE IF ERROR
033346  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
033350  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033352  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033356                          160$:

033356                          170$:
033356  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
033362  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
033364  000240                        NOP                     ;RETURN HERE IF ERROR
033366  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
033370  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033372  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033376                          180$:

                                ;CHANGE LOOP ADDRESSES
033376  012737  033414  001122        MOV     #190$,$LPADR
033404  012737  033414  001124        MOV     #190$,$LPERR
033412  000410                        BR      200$                    ;SKIP TO NEXT OPERATION

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

033414                          190$:

                                ;PREPARE DEVICE FOR READ OPERATION
033414  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
033420  054130                        .WORD   054130  ;TASK DESCRIPTOR
033422  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
033424  000240                        NOP                     ;RETURN HERE IF ERROR
033426  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033430  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
8183  033434                                  200$:
8184
8185  033434                                  240$:
8186
8187                                          ;READ DATA FROM DEVICE
8188  033434  012737  000051  001400                MOV     #WCD!GO,RMCS1O          ;READ DATA COMMAND
8189  033442  012702  001535                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8190  033446  112722  000006                        MOVB    #RMDA,(R2)+
8191  033452  112722  000032                        MOVB    #RMOF,(R2)+
8192  033456  112722  000034                        MOVB    #RMDC,(R2)+
8193  033462  112722  000004                        MOVB    #RMBA,(R2)+
8194  033466  112722  000002                        MOVB    #RMWC,(R2)+
8195  033472  112722  000000                        MOVB    #RMCS1,(R2)+
8196  033476  112712  000200                        MOVB    #200,(R2)
8197
8198  033502  004737  044006                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8199  033506  000404                                 BR      250$                    ;GO TO 250$ IF NO ERROR
8200  033510  000240                                 NOP                     ;RETURN HERE IF ERROR
8201  033512  104000                                 ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8202  033514  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8203  033520                                  250$:
8204
8205                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8206  033520  004737  043452                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8207
8208                                          ;WAIT FOR READ OPERATION TO COMPLETE
8209  033524  004737  044346                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8210
8211                                          ;GO READ STATUS FOR READ OPERATION
8212  033530  004737  043536                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8213  033534  000404                                 BR      260$                    ;GO TO 260$ IF NO ERROR
8214  033536  000240                                 NOP                     ;RETURN HERE IF ERROR
8215  033540  104000                                 ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8216  033542  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8217  033546                                  260$:
8218
8219                                          ;CHECK FOR ERRORS DURING READ OPERATION
8220  033546  004737  044532                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8221  033552  000405                                 BR      270$                    ;GO TO 270$ IF NO ERROR
8222  033554  000240                                 NOP                     ;RETURN HERE IF ERROR
8223  033556  104000                                 ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8224  033560  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8225  033562  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8226  033566                                  270$:
8227  033566  004737  057036                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8228  033572  000405                                 BR      280$                    ;GO TO 280$ IF NO ERROR
8229  033574  000240                                 NOP                     ;RETURN HERE IF ERROR
8230  033576  104000                                 ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8231  033600  004736                                 JSR     PC,@(SP)+       ,GO BACK FOR MORE ERROR CHECKS
8232  033602  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8233  033606                                  280$:
8234
8235  033606                                  290$:
8236  033606  004737  045364                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8237  033612  000405                                 BR      300$                    ;GO TO 300$ IF NO ERROR
8238  033614  000240                                 NOP                     ;RETURN HERE IF ERROR
```

```
8239  033616  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8240  033620  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8241  033622  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8242  033626                        300$:
8243
8244  033626                        350$:
8245                                ;;**************************************************************
8246                                ;*TEST 25        WRITE, READ EARLY PEAK SHIFT
8247                                ;;**************************************************************
8248  033626                        TST25:
8249  033626  000004                        SCOPE                   ;SCOPE CALL
8250  033630  000240                        NOP                     ;START OF TEST
8251  033632  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8252  033636  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8253  033642  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8254  033646  012737  000025  001226        MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8255
8256                                ;;**************************************************************
8257                                ;LOOP #1        FORMAT,WRITE,READ
8258
8259  033654                        10$:
8260
8261                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
8262  033654  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8263  033660  054130                        .WORD   054130  ;TASK DESCRIPTOR
8264  033662  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
8265  033664  000240                        NOP                     ;RETURN HERE IF ERROR
8266  033666  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8267  033670  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8268  033674                        20$:
8269
8270                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8271  033674  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
8272  033702  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
8273  033710  012737  107546  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8274  033716  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
8275  033724  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8276  033732  012737  000063  001400        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8277
8278                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8279
8280  033740  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8281  033744  000405                        BR      25$                     ;GO TO 25$ IF NO ERROR
8282  033746  104401  067464                TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8283  033752  104000                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8284  033754  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8285  033760                        25$:
8286  033760  012737  071320  001174        MOV     #EARLY,$TMP0            ;STARTING ADDRESS OF PATTERN
8287  033766  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
8288  033774  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8289
8290                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8291  034000  012702  001535                MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8292  034004  112722  000034                MOVB    #RMDC,(R2)+
8293  034010  112722  000006                MOVB    #RMDA,(R2)+
8294  034014  112722  000004                MOVB    #RMBA,(R2)+
```

```
8295  034020  112722  000002          MOVB    #RMWC,(R2)+
8296  034024  112722  000032          MOVB    #RMOF,(R2)+
8297  034030  112722  000000          MOVB    #RMCS1,(R2)+
8298  034034  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
8299  034040                      30$:
8300
8301                              ;FORMAT THE DRIVE
8302  034040  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8303  034044  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
8304  034046  000240                  NOP                     ;RETURN HERE IF ERROR
8305  034050  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8306  034052  000137  034650          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8307  034056                      40$:
8308
8309                              ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8310  034056  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8311
8312                              ;WAIT FOR THE FORMAT TO COMPLETE
8313  034062  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8314
8315                              ;GO READ STATUS FOR FORMAT OPERATION
8316  034066  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8317  034072  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
8318  034074  000240                  NOP                     ;RETURN HERE IF ERROR
8319  034076  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8320  034100  000137  034650          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8321  034104                      50$:
8322
8323                              ;VERIFY NO EPRORS DURING FORMAT
8324  034104  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8325  034110  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
8326  034112  000240                  NOP                     ;RETURN HERE IF ERROR
8327  034114  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8328  034116  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8329  034120  000137  034650          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8330  034124                      60$:
8331
8332                              ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8333  034124  012737  034142  001122      MOV     #70$,$LPADR
8334  034132  012737  034142  001124      MOV     #70$,$LPERR
8335  034140  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
8336
8337                              ;;*****************************************************************
8338                              ;LOOP #2        WRITE,READ
8339
8340  034142                      70$:
8341
8342
8343                              ;PREPARE DEVICE FOR WRITE OPERATION
8344  034142  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8345  034146  054130                  .WORD   054130  ;TASK DESCRIPTOR
8346  034150  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
8347  034152  000240                  NOP                     ;RETURN HERE IF ERROR
8348  034154  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8349  034156  000137  034650          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8350  034162                      80$:
```

```
8351
8352
8353  034162                          120$:
8354
8355                                  ;WRITE DATA TO THE DRIVE
8356  034162  012737  177400  001402          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
8357  034170  012737  107552  001404          MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
8358  034176  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
8359  034204  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8360  034210  112722  000006                  MOVB    #RMDA,(R2)+
8361  034214  112722  000034                  MOVB    #RMDC,(R2)+
8362  034220  112722  000032                  MOVB    #RMOF,(R2)+
8363  034224  112722  000004                  MOVB    #RMBA,(R2)+
8364  034230  112722  000002                  MOVB    #RMWC,(R2)+
8365  034234  112722  000000                  MOVB    #RMCS1,(R2)+
8366  034240  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
8367
8368  034244  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8369  034250  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
8370  034252  000240                          NOP                     ;RETURN HERE IF ERROR
8371  034254  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8372  034256  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8373  034262                          130$:
8374
8375                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8376  034262  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8377
8378                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
8379  034266  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8380
8381                                  ;GO READ STATUS FOR WRITE COMMAND
8382  034272  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8383  034276  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
8384  034300  000240                          NOP                     ;RETURN HERE IF ERROR
8385  034302  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8386  034304  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8387  034310                          140$:
8388
8389                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
8390  034310  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
8391  034314  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8392  034316  000240                          NOP                     ;RETURN HERE IF ERROR
8393  034320  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8394  034322  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8395  034324  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8396  034330                          150$:
8397  034330  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8398  034334  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
8399  034336  000240                          NOP                     ;RETURN HERE IF ERROR
8400  034340  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8401  034342  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8402  034344  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8403  034350                          160$:
8404
8405  034350                          170$:
8406  034350  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
```

```
8407  034354  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
8408  034356  000240                          NOP                     ;RETURN HERE IF ERROR
8409  034360  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8410  034362  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8411  034364  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8412  034370                          180$:
8413
8414                                  ;CHANGE LOOP ADDRESSES
8415  034370  012737  034406  001122          MOV     #190$,$LPADR
8416  034376  012737  034406  001124          MOV     #190$,$LPERR
8417  034404  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
8418
8419                                  ;;*****************************************************************
8420                                  ;LOOP #3        READ
8421
8422  034406                          190$:
8423
8424                                  ;PREPARE DEVICE FOR READ OPERATION
8425  034406  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8426  034412  054130                          .WORD   054130  ;TASK DESCRIPTOR
8427  034414  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
8428  034416  000240                          NOP                     ;RETURN HERE IF ERROR
8429  034420  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8430  034422  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8431  034426                          200$:
8432
8433  034426                          240$:
8434
8435                                  ;READ DATA FROM DEVICE
8436  034426  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8437  034434  012737  110556  001404          MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
8438  034442  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8439  034446  112722  000006                  MOVB    #RMDA,(R2)+
8440  034452  112722  000032                  MOVB    #RMOF,(R2)+
8441  034456  112722  000034                  MOVB    #RMDC,(R2)+
8442  034462  112722  000004                  MOVB    #RMBA,(R2)+
8443  034466  112722  000002                  MOVB    #RMWC,(R2)+
8444  034472  112722  000000                  MOVB    #RMCS1,(R2)+
8445  034476  112712  000200                  MOVB    #200,(R2)
8446
8447  034502  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8448  034506  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
8449  034510  000240                          NOP                     ;RETURN HERE IF ERROR
8450  034512  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8451  034514  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8452  034520                          250$:
8453
8454                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8455  034520  004737  04345?                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8456
8457                                  ;WAIT FOR READ OPERATION TO COMPLETE
8458  034524  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8459
8460                                  ;GO READ STATUS FOR READ OPERATION
8461  034530  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8462  034534  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
```

```
 8463  034536  000240                        NOP                     ;RETURN HERE IF ERROR
 8464  034540  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8465  034542  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8466  034546                        260$:
 8467
 8468                                ;CHECK FOR ERRORS DURING READ OPERATION
 8469  034546  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8470  034552  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
 8471  034554  000240                        NOP                     ;RETURN HERE IF ERROR
 8472  034556  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8473  034560  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8474  034562  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8475  034566                        270$:
 8476  034566  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8477  034572  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
 8478  034574  000240                        NOP                     ;RETURN HERE IF ERROR
 8479  034576  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 8480  034600  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8481  034602  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8482  034606                        280$:
 8483
 8484  034606                        290$:
 8485  034606  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8486  034612  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
 8487  034614  000240                        NOP                     ;RETURN HERE IF ERROR
 8488  034616  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8489  034620  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8490  034622  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8491  034626                        300$:
 8492  034626  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
 8493  034632  107552                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
 8494  034634  110556                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
 8495  034636  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
 8496  034640  000240                        NOP                     ;RETURN HERE IF ERROR
 8497  034642  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 8498  034644  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8499  034650                        310$:
 8500
 8501  034650                        350$:
 8502                                ;;*****************************************************************
 8503                                ;*TEST 26        WRITE, READ MIXED PEAK SHIFT
 8504                                ;;*****************************************************************
 8505  034650                        TST26:
 8506  034650  000004                        SCOPE                   ;SCOPE CALL
 8507  034652  000240                        NOP                     ;START OF TEST
 8508  034654  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 8509  034660  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 8510  034664  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 8511  034670  012737  000026  001226        MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8512
 8513                                ;;*****************************************************************
 8514                                ;LOOP #1        FORMAT,WRITE,READ
 8515
 8516  034676                        10$:
 8517
 8518                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
```

```
8519  034676  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8520  034702  054130                  .WORD   054130  ;TASK DESCRIPTOR
8521  034704  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
8522  034706  000240                  NOP                     ;RETURN HERE IF ERROR
8523  034710  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8524  034712  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8525  034716                  20$:
8526
8527                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8528  034716  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
8529  034724  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
8530  034732  012737  107546  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8531  034740  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
8532  034746  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8533  034754  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8534
8535                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8536
8537  034762  004737  040734          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8538  034766  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
8539  034770  104401  067464          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8540  034774  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8541  034776  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8542  035002                  25$:
8543  035002  012737  071004  001174          MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
8544  035010  012737  000200  001176          MOV     #128.,$TMP1             ;RANGE OF PATTERN
8545  035016  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8546
8547                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8548  035022  012702  001535          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8549  035026  112722  000034          MOVB    #RMDC,(R2)+
8550  035032  112722  000006          MOVB    #RMDA,(R2)+
8551  035036  112722  000004          MOVB    #RMBA,(R2)+
8552  035042  112722  000002          MOVB    #RMWC,(R2)+
8553  035046  112722  000032          MOVB    #RMOF,(R2)+
8554  035052  112722  000000          MOVB    #RMCS1,(R2)+
8555  035056  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
8556  035062                  30$:
8557
8558                          ;FORMAT THE DRIVE
8559  035062  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8560  035066  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
8561  035070  000240                  NOP                     ;RETURN HERE IF ERROR
8562  035072  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8563  035074  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8564  035100                  40$:
8565
8566                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8567  035100  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8568
8569                          ;WAIT FOR THE FORMAT TO COMPLETE
8570  035104  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8571
8572                          ;GO READ STATUS FOR FORMAT OPERATION
8573  035110  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8574  035114  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
```

```
8575  035116  000240                          NOP                     ;RETURN HERE IF ERROR
8576  035120  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8577  035122  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8578  035126                          50$:
8579
8580                                  ;VERIFY NO ERRORS DURING FORMAT
8581  035126  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8582  035132  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
8583  035134  000240                          NOP                     ;RETURN HERE IF ERROR
8584  035136  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8585  035140  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8586  035142  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8587  035146                          60$:
8588
8589                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8590  035146  012737  035164  001122          MOV     #70$,$LPADR
8591  035154  012737  035164  001124          MOV     #70$,$LPERR
8592  035162  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
8593
8594                                  ;;*****************************************************************
8595                                  ;LOOP #2        WRITE,READ
8596
8597  035164                          70$:
8598      -
8599
8600                                  ;PREPARE DEVICE FOR WRITE OPERATION
8601  035164  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8602  035170  054130                          .WORD   054130  ;TASK DESCRIPTOR
8603  035172  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8604  035174  000240                          NOP                     ;RETURN HERE IF ERROR
8605  035176  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8606  035200  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8607  035204                          80$:
8608
8609
8610  035204                          120$:
8611
8612                                  ;WRITE DATA TO THE DRIVE
8613  035204  012737  177400  001402          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
8614  035212  012737  107552  001404          MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
8615  035220  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
8616  035226  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8617  035232  112722  000006                  MOVB    #RMDA,(R2)+
8618  035236  112722  000034                  MOVB    #RMDC,(R2)+
8619  035242  112722  000032                  MOVB    #RMOF,(R2)+
8620  035246  112722  000004                  MOVB    #RMBA,(R2)+
8621  035252  112722  000002                  MOVB    #RMWC,(R2)+
8622  035256  112722  000000                  MOVB    #RMCS1,(R2)+
8623  035262  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
8624
8625  035266  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8626  035272  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
8627  035274  000240                          NOP                     ;RETURN HERE IF ERROR
8628  035276  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8629  035300  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8630  035304                          130$:
```

```
8631
8632                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8633  035304  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8634
8635                                    ;WAIT FOR WRITE COMMAND TO COMPLETE
8636  035310  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8637
8638                                    ;GO READ STATUS FOR WRITE COMMAND
8639  035314  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8640  035320  000404                            BR      140$                    ;GO TO 140$ IF NO ERROR
8641  035322  000240                            NOP                     ;RETURN HERE IF ERROR
8642  035324  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8643  035326  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8644  035332                                140$:
8645
8646                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
8647  035332  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8648  035336  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
8649  035340  000240                            NOP                     ;RETURN HERE IF ERROR
8650  035342  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8651  035344  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8652  035346  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8653  035352                                150$:
8654  035352  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8655  035356  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
8656  035360  000240                            NOP                     ;RETURN HERE IF ERROR
8657  035362  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8658  035364  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8659  035366  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8660  035372                                160$:
8661
8662  035372                                170$:
8663  035372  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8664  035376  000405                            BR      180$                    ;GO TO 180$ IF NO ERROR
8665  035400  000240                            NOP                     ;RETURN HERE IF ERROR
8666  035402  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8667  035404  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8668  035406  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8669  035412                                180$:
8670
8671                                    ;CHANGE LOOP ADDRESSES
8672  035412  012737  035430  001122          MOV     #190$,$LPADR
8673  035420  012737  035430  001124          MOV     #190$,$LPERR
8674  035426  000410                            BR      200$                    ;SKIP TO NEXT OPERATION
8675
8676                                    ;;*****************************************************************
8677                                    ;LOOP #3        READ
8678
8679  035430                                190$:
8680
8681                                    ;PREPARE DEVICE FOR READ OPERATION
8682  035430  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8683  035434  054130                            .WORD   054130  ;TASK DESCRIPTOR
8684  035436  000404                            BR      200$                    ;GO TO 200$ IF NO ERROR
8685  035440  000240                            NOP                     ;RETURN HERE IF ERROR
8686  035442  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
8687  035444  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8688  035450                          200$:
8689
8690  035450                          240$:
8691
8692                                  ;READ DATA FROM DEVICE
8693  035450  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;CHANGE BUFFER
8694  035456  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8695  035464  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8696  035470  112722  000006                  MOVB    #RMDA,(R2)+
8697  035474  112722  000032                  MOVB    #RMOF,(R2)+
8698  035500  112722  000034                  MOVB    #RMDC,(R2)+
8699  035504  112722  000004                  MOVB    #RMBA,(R2)+
8700  035510  112722  000002                  MOVB    #RMWC,(R2)+
8701  035514  112722  000000                  MOVB    #RMCS1,(R2)+
8702  035520  112712  000200                  MOVB    #200,(R2)
8703
8704  035524  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8705  035530  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
8706  035532  000240                          NOP                     ;RETURN HERE IF ERROR
8707  035534  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8708  035536  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8709  035542                          250$:
8710
8711                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8712  035542  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8713
8714                                  ;WAIT FOR READ OPERATION TO COMPLETE
8715  035546  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8716
8717                                  ;GO READ STATUS FOR READ OPERATION
8718  035552  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8719  035556  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
8720  035560  000240                          NOP                     ;RETURN HERE IF ERROR
8721  035562  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8722  035564  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8723  035570                          260$:
8724
8725                                  ;CHECK FOR ERRORS DURING READ OPERATION
8726  035570  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
8727  035574  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
8728  035576  000240                          NOP                     ;RETURN HERE IF ERROR
8729  035600  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8730  035602  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8731  035604  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8732  035610                          270$:
8733  035610  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8734  035614  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
8735  035616  000240                          NOP                     ;RETURN HERE IF ERROR
8736  035620  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8737  035622  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8738  035624  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8739  035630                          280$:
8740
8741  035630                          290$:
8742  035630  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
```

```
8743  035634  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
8744  035636  000240                          NOP                     ;RETURN HERE IF ERROR
8745  035640  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8746  035642  004736                          JSP     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8747  035644  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8748  035650                          300$:
8749  035650  004737  043104                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
8750  035654  107552                          .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
8751  035656  110556                          .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
8752  035660  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
8753  035662  000240                          NOP                     ;RETURN HERE IF ERROR
8754  035664  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
8755  035666  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8756  035672                          310$:
8757
8758  035672                          350$:
8759                                  ;;*****************************************************************
8760                                  ;*TEST 27        WRITE, READ EACH TRACK
8761                                  ;;*****************************************************************
8762  035672                          TST27:
8763  035672  000004                          SCOPE                   ;SCOPE CALL
8764  035674  000240                          NOP                     ;START OF TEST
8765  035676  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8766  035702  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8767  035706  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8768  035712  012737  000027  001226          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8769
8770                                  ;;*****************************************************************
8771                                  ;LOOP #1        FORMAT,WRITE,READ
8772
8773  035720                          10$:
8774
8775                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
8776  035720  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8777  035724  054130                          .WORD   054130  ;TASK DESCRIPTOR
8778  035726  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
8779  035730  000240                          NOP                     ;RETURN HERE IF ERROR
8780  035732  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8781  035734  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8782  035740                          20$:
8783
8784                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8785  035740  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
8786  035746  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
8787  035754  012737  107546  001404  25$:    MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8788  035762  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
8789  035770  012737  010003  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8790  035776  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8791
8792                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8793
8794  036004  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8795  036010  000405                          BR      26$                     ;GO TO 26$ IF NO ERROR
8796  036012  104401  067464                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8797  036016  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8798  036020  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
8799  036024                                  26$:
8800  036024  012737  071004  001174                  MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
8801  036032  012737  000200  001176                  MOV     #128.,$TMP1             ;RANGE OF PATTERN
8802  036040  004737  042640                          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8803
8804                                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8805  036044  012702  001535                          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8806  036050  112722  000034                          MOVB    #RMDC,(R2)+
8807  036054  112722  000006                          MOVB    #RMDA,(R2)+
8808  036060  112722  000004                          MOVB    #RMBA,(R2)+
8809  036064  112722  000002                          MOVB    #RMWC,(R2)+
8810  036070  112722  000032                          MOVB    #RMOF,(R2)+
8811  036074  112722  000000                          MOVB    #RMCS1,(R2)+
8812  036100  112712  000200                          MOVB    #200,(R2)               ;TERMINATE TABLE
8813  036104                                  30$:
8814
8815                                          ;FORMAT THE DRIVE
8816  036104  004737  044006                          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8817  036110  000404                                  BR      40$                     ;GO TO 40$ IF NO ERROR
8818  036112  000240                                  NOP                     ;RETURN HERE IF ERROR
8819  036114  104000                                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8820  036116  000137  036736                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8821  036122                                  40$:
8822
8823                                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8824  036122  004737  043452                          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8825
8826                                          ;WAIT FOR THE FORMAT TO COMPLETE
8827  036126  004737  044346                          JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8828
8829                                          ;GO READ STATUS FOR FORMAT OPERATION
8830  036132  004737  043536                          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8831  036136  000404                                  BR      50$                     ;GO TO 50$ IF NO ERROR
8832  036140  000240                                  NOP                     ;RETURN HERE IF ERROR
8833  036142  104000                                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8834  036144  000137  036736                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8835  036150                                  50$:
8836
8837                                          ;VERIFY NO ERRORS DURING FORMAT
8838  036150  004737  057036                          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8839  036154  000405                                  BR      60$                     ;GO TO 60$ IF NO ERROR
8840  036156  000240                                  NOP                     ;RETURN HERE IF ERROR
8841  036160  104000                                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8842  036162  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8843  036164  000137  036736                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8844  036170                                  60$:
8845
8846                                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8847  036170  012737  036206  001122                  MOV     #70$,$LPADR
8848  036176  012737  036206  001124                  MOV     #70$,$LPERR
8849  036204  000410                                  BR      80$                     ;SKIP TO WRITE OPERATION
8850
8851                                          ;;*****************************************************************
8852                                          ;LOOP #2        WRITE,READ
8853
8854  036206                                  70$:
```

```
8855
8856
8857                                    ;PREPARE DEVICE FOR WRITE OPERATION
8858  036206  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8859  036212  054130                            .WORD   054130  ;TASK DESCRIPTOR
8860  036214  000404                            BR      80$                     ;GO TO 80$ IF NO ERROR
8861  036216  000240                            NOP                     ;RETURN HERE IF ERROR
8862  036220  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8863  036222  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8864  036226                                80$:
8865
8866
8867  036226                                120$:
8868
8869                                    ;WRITE DATA TO THE DRIVE
8870  036226  012737  107552  001404            MOV     #BUFONE+4,RMBAO         ;CHANGE BUS ADDRESS
8871  036234  012737  177400  001402            MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
8872  036242  012737  000061  001400            MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
8873  036250  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8874  036254  112722  000006                    MOVB    #RMDA,(R2)+
8875  036260  112722  000034                    MOVB    #RMDC,(R2)+
8876  036264  112722  000032                    MOVB    #RMOF,(R2)+
8877  036270  112722  000004                    MOVB    #RMBA,(R2)+
8878  036274  112722  000002                    MOVB    #RMWC,(R2)+
8879  036300  112722  000000                    MOVB    #RMCS1,(R2)+
8880  036304  112722  000200                    MOVB    #200,(R2)+              ;TERMINATE TABLE
8881
8882  036310  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8883  036314  000404                            BR      130$                    ;GO TO 130$ IF NO ERROR
8884  036316  000240                            NOP                     ;RETURN HERE IF ERROR
8885  036320  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8886  036322  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8887  036326                                130$:
8888
8889                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8890  036326  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8891
8892                                    ;WAIT FOR WRITE COMMAND TO COMPLETE
8893  036332  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8894
8895                                    ;GO READ STATUS FOR WRITE COMMAND
8896  036336  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8897  036342  000404                            BR      140$                    ;GO TO 140$ IF NO ERROR
8898  036344  000240                            NOP                     ;RETURN HERE IF ERROR
8899  036346  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8900  036350  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8901  036354                                140$:
8902
8903                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
8904  036354  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8905  036360  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
8906  036362  000240                            NOP                     ;RETURN HERE IF ERROR
8907  036364  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8908  036366  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8909  036370  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8910  036374                                150$:
```

```
8911  036374  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8912  036400  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
8913  036402  000240                  NOP                     ;RETURN HERE IF ERROR
8914  036404  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8915  036406  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8916  036410  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8917  036414                  160$:
8918
8919  036414                  170$:
8920  036414  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8921  036420  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
8922  036422  000240                  NOP                     ;RETURN HERE IF ERROR
8923  036424  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8924  036426  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8925  036430  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8926  036434                  180$:
8927
8928                          ;CHANGE LOOP ADDRESSES
8929  036434  012737  036452  001122          MOV     #190$,$LPADR
8930  036442  012737  036452  001124          MOV     #190$,$LPERR
8931  036450  000410                  BR      200$                    ;SKIP TO NEXT OPERATION
8932
8933                          ;;*****************************************************************
8934                          ;LOOP #3        READ
8935
8936  036452                  190$:
8937
8938                          ;PREPARE DEVICE FOR READ OPERATION
8939  036452  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8940  036456  054130                  .WORD   054130  ;TASK DESCRIPTOR
8941  036460  000404                  BR      200$                    ;GO TO 200$ IF NO ERROR
8942  036462  000240                  NOP                     ;RETURN HERE IF ERROR
8943  036464  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8944  036466  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8945  036472                  200$:
8946
8947  036472                  240$:
8948
8949                          ;READ DATA FROM DEVICE
8950  036472  012737  110556  001404          MOV     #BUFTWO+4,RMBAO         ;CHANGE BUS ADDRESS
8951  036500  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8952  036506  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8953  036512  112722  000006          MOVB    #RMDA,(R2)+
8954  036516  112722  000032          MOVB    #RMOF,(R2)+
8955  036522  112722  000034          MOVB    #RMDC,(R2)+
8956  036526  112722  000004          MOVB    #RMBA,(R2)+
8957  036532  112722  000002          MOVB    #RMWC,(R2)+
8958  036536  112722  000000          MOVB    #RMCS1,(R2)+
8959  036542  112712  000200          MOVB    #200,(R2)
8960
8961  036546  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8962  036552  000404                  BR      250$                    ;GO TO 250$ IF NO ERROR
8963  036554  000240                  NOP                     ;RETURN HERE IF ERROR
8964  036556  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8965  036560  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8966  036564                  250$:
```

```
8967
8968                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8969  036564  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8970
8971                                    ;WAIT FOR READ OPERATION TO COMPLETE
8972  036570  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8973
8974                                    ;GO READ STATUS FOR READ OPERATION
8975  036574  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8976  036600  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
8977  036602  000240                            NOP                     ;RETURN HERE IF ERROR
8978  036604  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8979  036606  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8980  036612                          260$:
8981
8982                                    ;CHECK FOR ERRORS DURING READ OPERATION
8983  036612  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8984  036616  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
8985  036620  000240                            NOP                     ;RETURN HERE IF ERROR
8986  036622  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8987  036624  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8988  036626  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8989  036632                          270$:
8990  036632  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8991  036636  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
8992  036640  000240                            NOP                     ;RETURN HERE IF ERROR
8993  036642  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8994  036644  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8995  036646  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8996  036652                          280$:
8997
8998  036652                          290$:
8999  036652  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
9000  036656  000405                            BR      300$                    ;GO TO 300$ IF NO ERROR
9001  036660  000240                            NOP                     ;RETURN HERE IF ERROR
9002  036662  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
9003  036664  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9004  036666  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
9005  036672                          300$:
9006  036672  004737  043104                    JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
9007  036676  107552                            .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
9008  036700  110556                            .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
9009  036702  000404                            BR      310$                    ;GO TO 310$ IF NO ERROR
9010  036704  000240                            NOP                     ;RETURN HERE IF ERROR
9011  036706  104000                            ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
9012  036710  000137  036736                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
9013  036714                          310$:
9014  036714  062737  000400  001406            ADD     #400,RMDAO              ;ADVANCE TO NEXT TRACK
9015  036722  022737  002037  001406            CMP     #2037,RMDAO             ;DONE??
9016  036730  103402                            BLO     350$                    ;YES!!
9017  036732  000137  035754                    JMP     25$                     ;TEST NEXT TRACK
9018
9019  036736                          350$:
9020  036736  012737  035720  001122            MOV     #10$,$LPADR             ;LOOP BACK TO START OF TEST
9021  036744  012737  035720  001124            MOV     #10$,$LPERR
9022                                    ;PUT NEWTEST HERE
```

```
9023  036752                                $EOSP:
9024  036752  000240                              NOP
9025  036754  013700  001450                      MOV     TSTQUE,R0
9026  036760  062700  000002                      ADD     #2,R0
9027  036764  010037  001450                      MOV     R0,TSTQUE
9028  036770  005710                              TST     (R0)
9029  036772  001402                              BEQ     1$
9030  036774  000137  007100                      JMP     READY
9031  037000  012737  001452  001450  1$:         MOV     #TSTQUE+2,TSTQUE
9032                                .SBTTL  END OF PASS ROUTINE
9033
9034                                ;;**********************************************************************
9035                                ;*INCREMENT THE PASS NUMBER ($PASS)
9036                                ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY''
9037                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
9038                                ;*IF THERES A MONITOR GO TO IT
9039                                ;*IF THERE ISN'T JUMP TO SHUT
9040
9041  037006                        $EOP:
9042  037006  000240                              NOP
9043  037010  005037  001116                      CLR     $TSTNM          ;;ZERO THE TEST NUMBER
9044  037014  005037  001206                      CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
9045  037020  005237  001230                      INC     $PASS           ;;INCREMENT THE PASS NUMBER
9046  037024  042737  100000  001230              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
9047  037032  005327                              DEC     (PC)+           ;;LOOP?
9048  037034  000001                $EOPCT: .WORD   1
9049  037036  003063                              BGT     $DOAGN          ;;YES
9050  037040  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
9051  037042  000001                $ENDCT: .WORD   1
9052  037044  037034                              $EOPCT
9053  037046  104401  037054                      TYPE    ,65$            ;;TYPE ASCIZ STRING
9054  037052  000407                              BR      64$             ;;GET OVER THE ASCIZ
9055                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
9056  037072                        64$:
9057  037072  013746  001230                      MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
9058                                                                        ;;TYPE PASS NUMBER
9059  037076  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
9060  037100  104401  037106                      TYPE    ,67$            ;;TYPE ASCIZ STRING
9061  037104  000421                              BR      66$             ;;GET OVER THE ASCIZ
9062                                ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
9063  037150                        66$:
9064  037150  013746  001126                      MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
9065                                                                        ;;TOTAL NUMBER OF ERRORS
9066  037154  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
9067  037156  104401  001217                      TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
9068  037162  005037  001126                      CLR     $ERTTL          ;;CLEAR ERROR TOTAL
9069  037166  013700  000042        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
9070  037172  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
9071  037174  000005                              RESET                   ;;CLEAR THE WORLD
9072  037176  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
9073  037200  000240                              NOP                     ;;SAVE ROOM
9074  037202  000240                              NOP                     ;;FOR
9075  037204  000240                              NOP                     ;;ACT11
9076  037206                        $DOAGN:
9077  037206  000137                              JMP     @(PC)+          ;;RETURN
9078  037210  063412                $RTNAD: .WORD   SHUT
```

```
 9079  037212      377      377      000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 9080           037216                            .EVEN
```

```
 9081                                   .SBTTL  SUBROUTINES
 9082
 9083                                   ;;*****************************************************************
 9084                                   .SBTTL  ERROR TYPEOUT ROUTINE
 9085
 9086                                   ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
 9087                                   ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
 9088                                   ;*
 9089                                   ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
 9090                                   ;*PRINTED ON THE FIRST LINE;
 9091                                   ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
 9092                                   ;*ONE OR MORE SUCCEEDING LINES;
 9093                                   ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
 9094                                   ;*ARE PRINTED AFTER THE ERROR MESSAGE.
 9095
 9096  037216                           ERRTYP:
 9097  037216  104414                          SAVREG
 9098  037220  032777  020000  141726          BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
 9099  037226  001402                          BEQ     1$              ;NO!!
 9100  037230  000137  037746                  JMP     21$             ;YES!!
 9101                                   ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
 9102  037234  104401  001217          1$:     TYPE    ,$CRLF
 9103  037240  104401  037762                  TYPE    ,ERTY00         ;TYPE "UNT#"
 9104  037244  013746  001234                  MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
 9105                                                                  ;;TYPE UNIT NUMBER
 9106  037250  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 9107  037252     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 9108  037253     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 9109  037254  005037  037752                  CLR     TSTNMB          ;LOAD TEST NUMBER FOR
 9110  037260  013737  001226  037752          MOV     $TESTN,TSTNMB
 9111  037266  104401  037770                  TYPE    ,ERTY01         ;TYPE "TST#"
 9112  037272  013746  037752                  MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
 9113                                                                  ;;TYPE TEST NUMBER
 9114  037276  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 9115  037300     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 9116  037301     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 9117  037302  005037  037754                  CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
 9118  037306  113737  001130  037754          MOVB    $ITEMB,ERRNMB   ;TYPEOUT
 9119  037314  001406                          BEQ     2$              ;SKIP IF NO ERROR CALLED
 9120  037316  104401  040000                  TYPE    ,ERTY02         ;TYPE "ERR#"
 9121  037322  013746  037754                  MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
 9122                                                                  ;;TYPE ERROR NUMBER
 9123  037326  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 9124  037330     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 9125  037331     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 9126  037332  104401  040007          2$:     TYPE    ,ERTY03         ;TYPE "PC="
 9127  037336  013746  001132                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 9128                                                                  ;;TYPE PROGRAM COUNTER
 9129  037342  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 9130  037344     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
 9131  037345     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 9132                                   ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
 9133  037346  005737  037754          3$:     TST     ERRNMB          ;WAS AN ERROR CALLED?
 9134  037352  001575                          BEQ     21$             ;NO!!
 9135  037354  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
 9136  037360  105037  037760                  CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
 9137  037364  105037  037761                 CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
 9138  037370  013700  037754                 MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
 9139  037374  006300                         ASL     R0              ;FOUR ENTRIES IN ERROR
 9140  037376  006300                         ASL     R0              ;TABLE
 9141  037400  006300                         ASL     R0
 9142  037402  062700  001554                 ADD     #$ERRTB-8.,R0
 9143  037406  011001                         MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
 9144                                                                 ;TABLE
 9145  037410  001507                         BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
 9146                                 ;TYPE THE ERROR MESSAGE
 9147  037412  012102                 4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
 9148  037414  001505                         BEQ     12$             ;BRANCH IF END OF MESSAGE
 9149  037416  010237  037564                 MOV     R2,11$          ;LOAD ADDRESS OF STRING
 9150  037422  005037  037756                 CLR     BOTADR          ;CLEAR BOT ADDRESS
 9151  037426  112203                 5$:     MOVB    (R2)+,R3        ;END OF STRING??
 9152  037430  001454                         BEQ     10$             ;YES!!
 9153  037432  122703  000015                 CMPB    #CR,R3          ;CARRIAGE RETURN??
 9154  037436  001003                         BNE     6$              ;NO!!
 9155  037440  105037  037761                 CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
 9156  037444  000770                         BR      5$              ;GET NEXT CHARACTER
 9157  037446  122703  000012         6$:     CMPB    #LF,R3          ;LINE FEED??
 9158  037452  001765                         BEQ     5$              ;YES-GET NEXT CHARACTER
 9159  037454  122703  000011                 CMPB    #HT,R3          ;HORIZONTAL TAB??
 9160  037460  001007                         BNE     8$              ;NO!!
 9161  037462  105237  037761         7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
 9162  037466  132737  000007  037761         BITB    #7,CHRCNT
 9163  037474  001372                         BNE     7$
 9164  037476  000407                         BR      9$
 9165  037500  105237  037761         8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
 9166  037504  122703  000040                 CMPB    #' ,R3          ;SPACE??
 9167  037510  001002                         BNE     9$              ;NO!!
 9168  037512  010237  037756                 MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
 9169  037516  122737  000100  037761 9$:     CMPB    #64.,CHRCNT     ;END OF LINE??
 9170  037524  103340                         BHIS    5$              ;NO!!
 9171  037526  013704  037756                 MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
 9172  037532  001007                         BNE     90$             ;BRANCH IF SPACE DETECTED
 9173  037534  104401  001217                 TYPE    ,$CRLF          ;TYPE CRLF
 9174  037540  105037  037761                 CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9175  037544  013702  037564                 MOV     11$,R2          ;SET UP R2 FOR TESTING
 9176  037550  000726                         BR      5$
 9177  037552  105044                 90$:    CLRB    -(R4)           ;REPLACE SPACE
 9178  037554  112737  177777  037760         MOVB    #-1,BOTFLG      ;SET BOT FLAG
 9179  037562  104401                 10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
 9180  037564  000000                 11$:    .WORD                   ;STRING ADDRESS GOES HERE
 9181  037566  105737  037760                 TSTB    BOTFLG          ;WAS STRING TRUNCATED??
 9182  037572  001707                         BEQ     4$              ;NO!!
 9183  037574  104401  001217                 TYPE    ,$CRLF          ;YES-TYPE CRLF
 9184  037600  105037  037760                 CLRB    BOTFLG          ;CLEAR BOT FLAG
 9185  037604  105037  037761                 CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9186  037610  013702  037756                 MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
 9187  037614  010237  037564                 MOV     R2,11$          ;SETUP 11$ FOR TYPING
 9188  037620  112742  000040                 MOVB    #' ,-(R2)       ;RESTORE SPACE
 9189  037624  105722                         TSTB    (R2)+           ;RESTORE R2
 9190  037626  000677                         BR      5$              ;TYPE REST OF STRING
 9191  037630                         12$:
 9192                                 ;TYPE ERROR HEADER AND ERROR DATA
```

```
9193  037630                          13$:
9194  037630  016001  000002                  MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
9195  037634  001444                          BEQ     21$             ;BRANCH IF NO HEADER
9196  037636  104401  001217                  TYPE    ,$CRLF          ;(ASSUME NO DATA)
9197  037642  016002  000004                  MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
9198  037646  016003  000006                  MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
9199  037652  012137  037662          14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
9200  037656  001433                          BEQ     21$             ;BRANCH IF END OF HEADERS
9201                                                                  ;(ASSUME END OF DATA)
9202  037660  104401                          TYPE
9203  037662  000000          15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
9204  037664  104401  001217                  TYPE    ,$CRLF
9205  037670  005702                          TST     R2              ;DATA WITH HEADER??
9206  037672  001767                          BEQ     14$             ;NO!!
9207  037674  012204                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
9208  037676  012305                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
9209  037700  105725          16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
9210  037702  100407                          BMI     18$             ;BINARY
9211  037704  001403                          BEQ     17$             ;OCTAL
9212  037706  013446                          MOV     @(R4)+,-(SP)    ;DECIMAL
9213  037710  104405                          TYPDS
9214  037712  000405                          BR      19$
9215  037714  013446          17$:    MOV     @(R4)+,-(SP)
9216  037716  104402                          TYPOC
9217  037720  000402                          BR      19$
9218  037722  013446          18$:    MOV     @(R4)+,-(SP)
9219  037724  104406                          TYPBN
9220  037726  005714          19$:    TST     (R4)            ;MORE DATA??
9221  037730  001403                          BEQ     20$             ;NO!!
9222  037732  104401  040015                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
9223  037736  000760                          BR      16$             ;AND CONTINUE
9224  037740  104401  001217          20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
9225  037744  000742                          BR      14$             ;BEFORE NEXT HEADER
9226  037746  104415          21$:    RESREG
9227  037750  000207                          RTS     PC
9228
9229  037752  000000          TSTNMB: .WORD   0               ;TEST NUMBER
9230  037754  000000          ERRNMB: .WORD   0               ;ERROR NUMBER
9231  037756  000000          BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
9232  037760     000          BOTFLG: .BYTE   0               ;BOT FLAG
9233  037761     000          CHRCNT: .BYTE   0               ;CHARACTER COUNT
9234
9235  037762  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
9236  037770  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
9237  037776  000043
9238  040000  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
9239  040006     000
9240  040007     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
9241  040014     000
9242  040015     040  000040          ERTY04: .ASCIZ  @ @
9243                                  .EVEN
9244
```

```
                        .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!!
```

```
9301  040112  004737  052700                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
9302  040116  000411                          BR      60$             ;CONTINUE - NO ERROR
9303  040120  000401                          BR      50$
9304  040122  000000                  40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
9305  040124  062716  000004          50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9306  040130  113776  040122  000000          MOVB    40$,@(SP)
9307  040136  000137  040720                  JMP     400$
9308  040142                          60$:
9309                                  ;;**************************************************************
9310                                  ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
9311  040142  032737  000100  040732          BIT     #BIT6,500$      ;VERIFY??
9312  040150  001433                          BEQ     120$            ;NO!!
9313  040152  004737  043452                  JSR     PC,GETSTS       ;SETUP INDEX TABLE
9314  040156  004737  043536                  JSR     PC,GET          ;GO GET STATUS
9315  040162  000411                          BR      90$             ;NO ERROR GETTING STATUS
9316  040164  000401                          BR      80$
9317  040166  000000                  70$:    .WORD                   ;ERROR FROM GETTING STATUS
9318  040170  062716  000004          80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9319  040174  113776  040166  000000          MOVB    70$,@(SP)
9320  040202  000137  040720                  JMP     400$
9321  040206  004737  053016          90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
9322  040212  000412                          BR      120$            ;NO ERROR IN CLEAR
9323  040214  000401                          BR      110$
9324  040216  000000                  100$:   .WORD                   ;ERROR IN STATUS CLEAR
9325  040220  005726                  110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
9326  040222  062716  000004                  ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
9327  040226  113776  040216  000000          MOVB    100$,@(SP)      ;ERROR TO USER
9328  040234  000137  040720                  JMP     400$
9329  040240                          120$:
9330
9331                                  ;;**************************************************************
9332                                  ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
9333                                  ;NOT VALID
9334  040240  032737  010000  040732          BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
9335  040246  001511                          BEQ     240$            ;NO!!
9336  040250  112737  000012  001506          MOVB    #RMDS,GETINX    ;GET RMDS
9337  040256  112737  000200  001507          MOVB    #200,GETINX+1
9338  040264  004737  043536                  JSR     PC,GET
9339  040270  000411                          BR      150$            ;NO ERROR GETTING RMDS
9340  040272  000401                          BR      140$
9341  040274  000000                  130$:   .WORD
9342  040276  062716  000004          140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9343  040302  113776  040274  000000          MOVB    130$,@(SP)
9344  040310  000137  040720                  JMP     400$
9345  040314  032737  000100  001342  150$:   BIT     #VV,RMDS1       ;IS VOLUME VALID??
9346  040322  001063                          BNE     240$            ;YES!!
9347  040324  005037  001474                  CLR     MEDENB          ;CLEAR MEDIA ENABLE
9348  040330  012737  000023  001400          MOV     #PAKACK!GO,RMCS10       ;LOAD PACK ACK COMMAND
9349  040336  112737  000000  001535          MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
9350  040344  112737  000200  001536          MOVB    #200,PUTINX+1
9351  040352  004737  044006                  JSR     PC,PUT          ;GO WRITE COMMAND
9352  040356  000410                          BR      180$            ;NO ERROR LOADING REGISTER
9353  040360  000401                          BR      170$
9354  040362  000000                  160$:   .WORD                   ;ERROR FROM PUT SUB
9355  040364  062716  000004          170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9356  040370  113776  040362  000000          MOVB    160$,@(SP)
```

```
9357  040376  000550                          BR      400$
9358  040400                          180$:
9359
9360                                  ;;*****************************************************************
9361                                  ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
9362  040400  032737  000020  040732          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
9363  040406  001431                          BEQ     240$            ;NO!!
9364  040410  004737  043452                  JSR     PC,GETSTS       ;SETUP FOR STATUS
9365  040414  004737  043536                  JSR     PC,GET          ;GO GET STATUS
9366  040420  000410                          BR      210$            ;NO ERROR GETTING STATUS
9367  040422  000401                          BR      200$
9368  040424  000000                  190$:   .WORD                   ;ERROR FROM GET SUB
9369  040426  062716  00000'          200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9370  040432  113776  040424  000000          MOVB    190$,@(SP)
9371  040440  000527                          BR      400$
9372  040442  004737  053676          210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
9373  040446  000411                          BR      240$            ;NO ERROR
9374  040450  000401                          BR      230$
9375  040452  000000                  220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
9376  040454  005726                  230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
9377  040456  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9378  040462  113776  040452  000000          MOVB    220$,@(SP)
9379  040470  000513                          BR      400$
9380  040472                          240$:
9381
9382                                  ;;*****************************************************************
9383                                  ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
9384  040472  032737  004000  040732          BIT     #BIT11,500$     ;RECALIBRATE??
9385  040500  001513                          BEQ     410$            ;NO!!
9386  040502  112737  000012  001506          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
9387  040510  112737  000200  001507          MOVB    #200,GETINX+1
9388  040516  004737  043536                  JSR     PC,GET          ;GO GET RMDS
9389  040522  000410                          BR      270$            ;NO ERROR GETTING RMDS
9390  040524  000401                          BR      260$
9391  040526  000000                  250$:   .WORD                   ;ERROR FROM GET SUB
9392  040530  062716  000004          260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9393  040534  113776  040526  000000          MOVB    250$,@(SP)
9394  040542  000466                          BR      400$
9395  040544  032737  020000  001342  270$:   BIT     #PIP,RMDS1      ;IS PIP ACTIVE??
9396  040552  001466                          BEQ     410$            ;NO!!
9397  040554  012737  000007  001400          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
9398  040562  112737  000000  001535          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
9399  040570  112737  000200  001536          MOVB    #200,PUTINX+1
9400  040576  004737  044006                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
9401  040602  000410                          BR      300$            ;NO ERROR
9402  040604  000401                          BR      290$
9403  040606  000000                  280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
9404  040610  062716  000004          290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9405  040614  113776  040606  000000          MOVB    280$,@(SP)
9406  040622  000436                          BR      400$
9407  040624  004737  043452          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
9408  040630  004737  044346                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
9409
9410                                  ;;*****************************************************************
9411                                  ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
9412  040634  032737  000010  040732          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 9413  040642  001432                        BEQ     410$            ;NO !
 9414  040644  004737  043536                JSR     PC,GET          ;GO GET STATUS
 9415  040650  000410                        BR      330$            ;NO ERROR GETTING STATUS
 9416  040652  000401                        BR      320$
 9417  040654  000000              310$:     .WORD                   ;ERROR FROM GET
 9418  040656  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9419  040662  113776  040654  000000        MOVB    310$,@(SP)
 9420  040670  000413                        BR      400$
 9421  040672  004737  054472      330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 9422  040676  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
 9423  040700  000401                        BR      350$
 9424  040702  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
 9425  040704  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
 9426  040706  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9427  040712  113776  040702  000000        MOVB    340$,@(SP)
 9428  040720  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 9429  040724  000240                        NOP
 9430  040726  000240                        NOP
 9431  040730  000207              410$:     RTS     PC              ;RETURN TO USER
 9432
 9433  040732  000000              500$:     .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  107546  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
```

```
9490  041014  012737  177376  001402          MOV     #<^C258.+1>,RMWCO       ;WORD COUNT
9491  041022  012737  010000  001432          MOV     #FMT16,RMOFO    ;16 BIT FORMAT
9492  041030  012737  ''5526  001404          MOV     #MFGFIL,RMBAO   ;BUFFER ADDRESS
9493
9494  041036  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
9495  041042  112720  000006                  MOVB    #RMDA,(R0)+
9496  041046  112720  000034                  MOVB    #RMDC,(R0)+
9497  041052  112720  000002                  MOVB    #RMWC,(R0)+
9498  041056  112720  000032                  MOVB    #RMOF,(R0)+
9499  041062  112720  000004                  MOVB    #RMBA,(R0)+
9500  041066  112720  000000                  MOVB    #RMCS1,(R0)+
9501  041072  112720  000200                  MOVB    #200,(R0)+
9502  041076  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9503
9504                          ;SET GET INDEX TABLE FOR READING STATUS
9505  041100  004737  043452                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
9506  041104                  30$:
9507
9508                          ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
9509  041104  012737  000011  001400          MOV     #DRVCLR!GO,RMCS10       ;LOAD COMMAND IN PUT BUFFER
9510  041112  004737  044006                  JSR     PC,PUT          ;OUTPUT COMMAND
9511  041116  000411                          BR      45$             ;RETURN HERE IF NO ERROR
9512  041120  000401                          BR      40$             ;GET AROUND ERROR #
9513  041122  000000          35$:    .WORD                           ;ERROR # GOES HERE
9514
9515  041124  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9516  041130  113776  041122  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
9517  041136  000137  041636                  JMP     215$
9518  041142  004737  043536          45$:    JSR     PC,GET          ;GO GET STATUS
9519  041146  000411                          BR      60$             ;RETURN HERE IF NO ERROR
9520  041150  000401                          BR      55$             ;GET AROUND ERROR #
9521  041152  000000          50$:    .WORD                           ;ERROR # GOES HERE
9522
9523  041154  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9524  041160  113776  041152  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
9525  041166  000137  041636                  JMP     215$
9526
9527  041172  004737  056234          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
9528  041176  000412                          BR      75$             ;RETURN HERE IF NO ERROR
9529  041200  000401                          BR      70$             ;GET AROUND ERROR
9530  041202  000000          65$:    .WORD                           ;ERROR # GOES HERE
9531
9532  041204  005726          70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
9533  041206  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9534  041212  113776  041202  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
9535  041220  000137  041636                  JMP     215$
9536
9537  041224          75$:
9538
9539                          ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
9540  041224  032737  000100  001342          BIT     #VV,RMDSI       ;IS VV RESET ??
9541  041232  001050                          BNE     120$            ;NO !!
9542  041234  012737  000023  001400          MOV     #PACACK!GO,RMCS10       ;LOAD COMMAND
9543  041242  004737  044006                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
9544  041246  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
9545  041250  000401                          BR      85$             ;GET AROUND ERROR #
```

```
9546  041252  000000                  80$:    .WORD                   ;ERROR # GOES HERE
9547
9548  041254  062716  000006          85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9549  041260  113776  041252  000000          MOVB    80$,@(SP)       ;MOVE ERROR # TO ERROR CALL
9550  041266  000137  041636                  JMP     215$
9551  041272  004737  043536          90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
9552  041276  000411                          BR      105$            ;RETURN HERE IF NO ERROR
9553  041300  000401                          BR      100$            ;GET AROUND ERROR #
9554  041302  000000                  95$:    .WORD                   ;ERROR # GOES HERE
9555
9556  041304  062716  000006          100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9557  041310  113776  041302  000000          MOVB    95$,@(SP)       ;MOVE ERROR # TO CALL
9558  041316  000137  041636                  JMP     215$
9559
9560  041322  004737  053676          105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
9561  041326  000412                          BR      120$            ;RETURN HERE IF NO ERROR
9562  041330  000401                          BR      115$            ;GET AROUND ERROR #
9563  041332  000000                  110$:   .WORD                   ;ERROR # GOES HERE
9564
9565  041334  005726                  115$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
9566  041336  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9567  041342  113776  041332  000000          MOVB    110$,@(SP)      ;MOVE ERROR # TO USERS ERROR CALL
9568  041350  000137  041636                  JMP     215$
9569
9570  041354                          120$:
9571
9572                                  ;RECALIBRATE THE DRIVE IF PIP IS SET
9573  041354  032737  020000  001342          BIT     #PIP,RMDS1      ;IS PIP SET ??
9574  041362  001452                          BEQ     165$            ;NO !!
9575
9576  041364  012737  000007  001400          MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
9577  041372  004737  044006                  JSR     PC,PUT          ;PUT THE RECAL COMMAND
9578  041376  000411                          BR      135$            ;RETURN HERE IF NO ERROR
9579  041400  000401                          BR      130$            ;GET AROUND ERROR #
9580  041402  000000                  125$:   .WORD                   ;ERROR # GOES HERE
9581
9582  041404  062716  000006          130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9583  041410  113776  041402  000000          MOVB    125$,@(SP)      ;MOVE ERROR # TO USERS CALL
9584  041416  000137  041636                  JMP     215$
9585  041422                          135$:
9586  041422  004737  044346                  JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
9587  041426  004737  043536                  JSR     PC,GET          ;GO GET RECAL STATUS
9588  041432  000411                          BR      150$            ;RETURN HERE IF NO ERROR
9589  041434  000401                          BR      145$            ;GET AROUND ERROR #
9590  041436  000000                  140$:   .WORD                   ;ERROR # GOES HERE
9591
9592  041440  062716  000006          145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9593  041444  113776  041436  000000          MOVB    140$,@(SP)      ;MOVE ERROR TO USERS CALL
9594  041452  000137  041636                  JMP     215$
9595
9596  041456  004737  054472          150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
9597  041462  000412                          BR      165$            ;RETURN HERE IF NO ERROR
9598  041464  000401                          BR      160$            ;GET AROUND ERROR #
9599  041466  000000                  155$:   .WORD                   ;ERROR # GOES HERE
9600
9601  041470  005726                  160$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
```

```
9602  041472  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
9603  041476  113776  041466  000000    MOVB    155$,@(SP)       ;MOVE ERROR # TO USERS CALL
9604  041504  000137  041636            JMP     215$
9605
9606  041510                    165$:
9607
9608                              ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
9609
9610  041510  012737  000073  001400    MOV     #RH!GO,RMCS10    ;LOAD READ HEADER AND DATA COMMAND
9611  041516  004737  044006            JSR     PC,PUT           ;PUT COMMAND
9612  041522  000411                    BR      180$             ;RETURN HERE IF NO ERROR
9613  041524  000401                    BR      175$             ;GET AROUND ERROR #
9614  041526  000000            170$:   .WORD                    ;ERROR # GOES HERE
9615
9616  041530  062716  000006    175$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
9617  041534  113776  041526  000000    MOVB    170$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
96    041542  000137  041636            JMP     215$
9619
9620  041546  004737  044346    180$:   JSR     PC,TIMOUT        ;WAIT FOR READ OPERATION TO COMPLETE
9621  041552  004737  043536            JSR     PC,GET           ;GO GET STATUS FOR READ OPERATION
9622  041556  000411                    BR      195$             ;RETURN HERE IF NO ERROR
9623  041560  000401                    BR      190$             ;GET AROUND ERROR #
9624  041562  000000            185$:   .WORD                    ;ERROR # GOES HERE
9625
9626  041564  062716  000006    190$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
9627  041570  113776  041562  000000    MOVB    185$,@(SP)       ;MOVE ERROR # TO CALL
9628  041576  000137  041636            JMP     215$
9629
9630  041602  004737  057036    195$:   JSR     PC,DTASTS        ;GO VERIFY RESULTS OF READ OPERATION
9631  041606  000412                    BR      210$             ;RETURN HERE IF NO ERROR
9632  041610  000401                    BR      205$             ;GET AROUND ERROR #
9633  041612  000000            200$:   .WORD                    ;ERROR # GOES HERE
9634
9635  041614  005726    205$:   TST     (SP)+            ;STRIP RETURN ADDRESS TO SUBROUTINE
9636  041616  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
9637  041622  113776  041612  000000    MOVB    200$,@(SP)       ;MOVE ERROR # TO USERS CALL
9638  041630  000137  041636            JMP     215$
9639
9640  041634  000456            210$:   BR      240$
9641
9642  041636                    215$:
9643
9644                              ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
9645                              ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
9646
9647  041636  032737  010000  001342    BIT     #MOL,RMDSI       ;IS MEDIUM ON LINE ??
9648  041644  001446                    BEQ     230$             ;YES !!
9649
9650  041646  005337  042626            DEC     500$             ;DECREMENT RETRY COUNT
9651  041652  100037                    BPL     225$             ;RETRY IF COUNT NOT NEG
9652
9653                              ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
9654  041654  013746  001344            MOV     RMER11,-(SP)     ;GET ER1
9655  041660  042716  100720            BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
9656  041664  032737  000010  001372    BIT     #DPE,RMER21      ;WAS THER A DATA PARITY ERROR ??
9657  041672  001402                    BEQ     220$             ;NO !!
```

```
9658  041674  042716  000010              BIC     #PAR,(SP)          ;YES - CLEAR PAR
9659  041700  005726                  220$: TST     (SP)+              ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
9660  041702  001027                        BNE     230$               ;YES !!
9661  041704  013746  001372                MOV     RMER21,-(SP)       ;GET ER2
9662  041710  042726  100010                BIC     #BSE!DPE,(SP)+     ;CLEAR MEDIA ERRORS
9663  041714  001022                        BNE     230$               ;BRANCH IF NONMEDIA ERRORS DETECTED
9664
9665                                ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
9666                                ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
9667                                ;ANOTHER AREA ON THE LAST TRACK
9668
9669  041716  062737  000002  001406        ADD     #2,RMDAO           ;ADVANCE SECTOR ADDRESS
9670  041724  122737  000012  001406        CMPB    #10.,RMDAO         ;QUIT IF ALL MFG SECTORS HAVE BEEN
9671  041732  001413                        BEQ     230$               ;TRIED
9672  041734  122737  000040  001406        CMPB    #32.,RMDAO         ;QUIT IF ALL USER SECTORS HAVE
9673  041742  001407                        BEQ     230$               ;BEEN TRIED
9674  041744  012737  000003  042626        MOV     #3,500$            ;REINSTATE RETRY COUNT FOR THIS SECTOR
9675  041752  162716  000006            225$: SUB     #6,(SP)            ;MOVE SP BACK TO NO ERROR
9676  041756  000137  041104                JMP     30$                ;RETRY THE READ OPERATION
9677
9678  041762                            230$:
9679
9680                                ;THE BAD SECTOR FILE CANNOT BE READ
9681  041762  162716  000004                SUB     #4,(SP)            ;MOVE SP TO ERROR RETURN
9682  041766  000137  042622                JMP     410$               ;GO TO MODULE EXIT
9683
9684  041772                            240$:
9685
9686                                ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
9687                                ;       READ THE USER FILE IF THIS IS THE MFG FILE
9688                                ;       OR ELSE DONE
9689
9690  041772  122737  000011  001406        CMPB    #9.,RMDAO          ;WAS THE USER FILE READ ??
9691  042000  103451                        BLO     260$               ;YES - READ IS COMPLETE
9692
9693                                ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
9694                                ;       ELSE DUMMY THE BAD SECTOR FILE
9695  042002  032737  140000  105526        BIT     #MSE!USE,MFGFIL    ;ARE THE BAD SECTOR FLAGS OFF ??
9696  042010  001410                        BEQ     250$               ;YES - 144 IS DISABLED
9697
9698  042012  012737  002012  001406        MOV     #002012,RMDAO      ;READ THE USER FILE
9699  042020  012737  000003  042626        MOV     #3,500$            ;RELOAD THE RETRY COUNT FOR THIS SECTOR
9700  042026  000137  041104                JMP     30$                ;GO READ THE USER FILE
9701  042032                            250$:
9702
9703                                ;DUMMY THE BAD SECTOR FILES
9704  042032  010046                        MOV     R0,-(SP)           ;;PUSH R0 ON STACK
9705  042034  010146                        MOV     R1,-(SP)           ;;PUSH R1 ON STACK
9706  042036  012701  000374                MOV     #252.,R1           ;R1 = NUMBER OF ENTRIES IN FILES
9707  042042  012700  000014                MOV     #14,R0             ;R0 = ADDRESS INDEX TO FILE  STORAGE
9708  042046  012760  177777  105526  255$: MOV     #-1,MFGFIL(R0)     ;ENTER ALL ONES IN MFG FILE
9709  042054  012760  177777  106536        MOV     #-1,USRFIL(R0)     ;ENTER ALL ONES IN USER FILE
9710  042062  062700  000002                ADD     #2,R0              ;ADVANCE ADDRESS
9711  042066  005301                        DEC     R1                 ;DECREMENT COUNT
9712  042070  001366                        BNE     255$               ;CONTINUE IF NOT DONE
9713  042072  012701  000006                MOV     #6.,R1             ;CLEAR ID, SERIAL NUMBER ETC
```

```
9714  042076  005000                          CLR     R0
9715  042100  005060  105526          256$:   CLR     MFGFIL(R0)
9716  042104  005060  106536                  CLR     USRFIL(R0)
9717  042110  062700  000002                  ADD     #2,R0
9718  042114  005301                          DEC     R1
9719  042116  001370                          BNE     256$
9720
9721  042120  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
9722  042122  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9723  042124                          260$:
9724
9725                                  ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
9726  042124  012737  177777  001474          MOV     #-1,MEDENB
9727
9728  042132  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9729  042134  005000                          CLR     R0              ;R0 IS REGISTER INDEX
9730  042136  016060  107546  001400  265$:   MOV     BUFFER(R0),PUTBUF(R0)
9731  042144  062700  000002                  ADD     #2,R0           ;ADVANCE R0
9732  042150  022700  000046                  CMP     #46,R0          ;DONE ??
9733  042154  103370                          BHIS    265$
9734
9735  042156  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9736  042160                          270$:
9737
9738  042160                          300$:
9739
9740                                  ;;**************************************************************
9741                                  ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
9742                                  ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
9743                                  ;SECTOR IS IN EITHER FILE.
9744
9745                                  ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
9746  042160  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9747  042162  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
9748  042164  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
9749  042166  013737  001434  001476  MOV     RMDCO,ASNDC     ;LOAD REQUESTED CYLINDER, TRACK,
9750  042174  013737  001406  001500  MOV     RMDAO,ASNDA     ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
9751  042202  005002                  CLR     R2              ;R2 = NUMBER OF SECTORS
9752  042204  013700  001402          MOV     RMWCO,R0        ;R0 = WORD COUNT
9753  042210  005300                  DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
9754  042212  005100                  COM     R0
9755  042214  012701  000400          MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
9756  042220  032737  000002  001400  BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
9757  042226  001402                  BEQ     305$            ;NO !!
9758  042230  012701  000402          MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
9759  042234  020100          305$:   CMP     R1,R0           ;IS THERE A FULL SECTOR ??
9760  042236  101404                  BLOS    310$            ;YES !!
9761  042240  005700                  TST     R0              ;IS R0 ZERO ??
9762  042242  001405                  BEQ     315$            ;YES !!
9763  042244  005202                  INC     R2              ;INCREMENT FOR PARTIAL SECTOR
9764  042246  000403                  BR      315$
9765  042250  160100          310$:   SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
9766  042252  005202                  INC     R2              ;INCREMENT SECTOR COUNT
9767  042254  000767                  BR      305$
9768  042256  010237  042626  315$:   MOV     R2,500$         ;SAVE SECTOR COUNT
9769
```

```
9770  042262                          316$:
9771
9772                                  ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
9773                                  ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
9774                                  ;MORE THAN ONE.
9775
9776  042262  012737  105542  042636          MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
9777                                                                  ;FILE TO BASE ADDRESS STORAGE
9778  042270  013737  001476  042632  320$:   MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
9779  042276  013737  001500  042634          MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
9780  042304  013737  042626  042630          MOV     500$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
9781
9782                                  ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
9783                                  ;CYLINDER, TRACK AND SECTOR ADDRESS
9784  042312  013700  042636          325$:   MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
9785  042316  012701  000376                  MOV     #<<127.*4>/2>,R1        ;R1 = LENGTH OF FILE
9786  042322  060100                          ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
9787  042324  021037  042632          330$:   CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
9788  042330  001405                          BEQ     345$            ;YES !!
9789  042332  101002                          BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
9790  042334                          335$:
9791
9792                                  ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
9793                                  ;SO ADD DISPLACEMENT TO CURRENT POSITION
9794  042334  060100                          ADD     R1,R0
9795  042336  000410                          BR      350$
9796  042340                          340$:
9797
9798                                  ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
9799                                  ;FROM CURRENT POSITION
9800  042340  160100                          SUB     R1,R0
9801  042342  000406                          BR      350$
9802  042344                          345$:
9803
9804                                  ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
9805                                  ;THE COMPARING TRACK, AND SECTOR.
9806  042344  026037  000002  042634          CMP     2(R0),530$      ;ARE THEY EQUAL ??
9807  042352  001413                          BEQ     360$            ;YES !!
9808  042354  101371                          BHI     340$            ;BRANCH IF HIGHER
9809  042356  000766                          BR      335$            ;BRANCH IF LOWER
9810  042360                          350$:
9811
9812                                  ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
9813                                  ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
9814  042360  005701                          TST     R1              ;IS DISPLACEMENT ZERO ??
9815  042362  001440                          BEQ     370$            ;YES !!
9816  042364  006201                          ASR     R1              ;HALVE THE DISLPACEMENT
9817  042366  042701  000003                  BIC     #3,R1
9818  042372  020037  042636                  CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
9819  042376  103432                          BLO     370$            ;NO !!
9820  042400  000751                          BR      330$            ;CONTINUE SEARCH
9821  042402                          360$:
9822
9823                                  ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
9824                                  ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
9825  042402  105237  001500                  INCB    ASNDA           ;INCREMENT SECTOR
```

```
 9826  042406  122737  000037  001500          CMPB    #31.,ASNDA      ;SECTOR OK ??
 9827  042414  103022                          BHIS    365$            ;YES !!
 9828  042416  105037  001500                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
 9829  042422  105237  001501                  INCB    ASNDA+1
 9830  042426  122737  000004  001501          CMPB    #4,ASNDA+1      ;TRACK OK ??
 9831  042434  103012                          BHIS    365$            ;YES !!
 9832  042436  005037  001500                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
 9833  042442  005237  001476                  INC     ASNDC           ;INCREMENT CYLINDER
 9834  042446  022737  001466  001476          CMP     #822.,ASNDC     ;CYLINDER OK ??
 9835  042454  103002                          BHIS    365$            ;YES !!
 9836  042456  005037  001476                  CLR     ASNDC
 9837  042462  000677                  365$:   BR      316$            ;REPEAT SEARCH
 9838
 9839  042464                          370$:
 9840
 9841                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
 9842                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
 9843                                  ;IS NOT ZERO.
 9844
 9845  042464  005337  042630                  DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
 9846  042470  001432                          BEQ     380$            ;DONE IF ZERO
 9847  042472  105237  042634                  INCB    530$            ;INCREMENT THE COMPARING SECTOR
 9848  042476  122737  000037  042634          CMPB    #31.,530$       ;SECTOR OK ??
 9849  042504  103022                          BHIS    375$            ;YES !!
 9850  042506  105037  042634                  CLRB    530$            ;CLEAR SECTOR
 9851  042512  105237  042635                  INCB    530$+1          ;INCREMENT TRACK
 9852  042516  122737  000004  042635          CMPB    #4,530$+1       ;TRACK OK ??
 9853  042524  103012                          BHIS    375$            ;YES !!
 9854  042526  005037  042634                  CLR     530$            ;CLEAR SECTOR TRACK
 9855  042532  005237  042632                  INC     520$            ;INCREMENT CYLINDER
 9856  042536  022737  001466  042632          CMP     #822.,520$      ;CYLINDER OK ??
 9857  042544  103002                          BHIS    375$            ;YES !!
 9858  042546  005037  042632                  CLR     520$            ;CLEAR CYLINDER
 9859  042552  000137  042312          375$:   JMP     325$            ;SEARCH NEXT SECTOR
 9860
 9861  042556                          380$:
 9862
 9863                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
 9864                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
 9865                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
 9866
 9867  042556  022737  106552  042636          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
 9868  042564  001405                          BEQ     400$            ;DONE IF BASE = USER
 9869  042566  012737  106552  042636          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
 9870  042574  000137  042270          390$:   JMP     320$            ;SEARCH USER FILE
 9871
 9872
 9873  042600                          400$:
 9874
 9875                                  ;ASSIGN THE SECTOR AND RETURN TO USER
 9876  042600  013737  001476  001434          MOV     ASNDC,RMDCO     ;LOAD CYLINDER
 9877  042606  013737  001500  001406          MOV     ASNDA,RMDAO     ;LOAD TRACK AND SECTOR
 9878  042614  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 9879  042616  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9880  042620  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9881  042622  000240                  410$:   NOP
```

```
 9882  042624  000207                          RTS     PC
 9883
 9884                                  ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
 9885
 9886  042626  000000                  500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 9887  042630  000000                  510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
 9888  042632  000000                  520$:   .WORD   ;COMPARING CYLINDER
 9889  042634  000000                  530$:   .WORD   ;COMPARING TRACK AND SECTOR
 9890  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$:    BIT     #BIT1,RMCS10    ;WRITE HEADER & DATA??
042704  001450                 BEQ     25$             ;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
```

```
9947   043052  001711                  BEQ     10$             ;START NEXT SECTOR IF 0
9948   043054  005303                  DEC     R3              ;DECREMENT PATTERN COUNT
9949   043056  001765                  BEQ     30$             ;RESTART PATTERN IF 0
9950   043060  000770                  BR      40$             ;CONTINUE DATA PATTERN
9951
9952   043062                  50$:
9953   043062  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
9954   043064  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
9955   043066  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
9956   043070  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
9957   043072  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
9958   043074  000240                  NOP
9959   043076  000207                  RTS     PC
9960
9961   043100  000000          60$:    .WORD                   ;CYLINDER ADDRESS STORAGE
9962   043102  000000          65$:    .WORD                   ;TRACK, SECTOR ADDRESS STORAGE
```

```
 9963                                   .SBTTL  COMPARE BUFFER SUBROUTINE
 9964
 9965                                   ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
 9966                                   ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
 9967                                   ,AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
 9968                                   ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
 9969
 9970                                   ;CALL:
 9971                                   ;(1)    JSR     PC,CMPBUF
 9972                                   ;(2)    .WORD   WRITE BUFFER ADDRESS
 9973                                   ;       .WORD   READ BUFFER ADDRESS
 9974                                   ;(3)    BR      ??              RETURN HERE IF NO ERROR
 9975                                   ;(4)    NOP                     RETURN HERE IF ERROR
 9976                                   ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
 9977                                   ;(6)    ???
 9978
 9979  043104                           CMPBUF:
 9980  043104  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9981  043106  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 9982  043110  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 9983  043112  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 9984
 9985  043114  005037  043450                  CLR     150$            ;CLEAR CORRECTION FLAG
 9986                                   ;DETERMINE IF DATA SHOULD BE CORRECTED
 9987  043120  032737  100000  001344          BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
 9988  043126  001465                          BEQ     80$             ;NO !!
 9989  043130  032737  000100  001344          BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
 9990  043136  001061                          BNE     80$             ;YES !!
 9991  043140  033737  004000  001362          BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
 9992  043146  001055                          BNE     80$             ;NO !!
 9993  043150  032737  010000  001362          BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
 9994  043156  001451                          BEQ     80$             ;NO !!
 9995
 9996                                   ;CORRECT DATA USING ECC INFORMATION
 9997  043160  013700  001404                  MOV     RMBAO,R0        ;R0 = STARTING BUFFER ADDRESS
 9998  043164  013701  001374                  MOV     RMEC1I,R1       ;R1 = ECC POSITION
 9999  043170  052737  100000  043450          BIS     #BIT15,150$     ;SET CORRECTION FLAG
10000                                   ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
10001
10002  043176  022701  000020          10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
10003  043202  103005                          BHIS    20$             ;NO !!
10004  043204  162701  000020                  SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
10005  043210  062700  000002                  ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
10006  043214  000770                          BR      10$
10007  043216  012702  000001          20$     MOV     #1,R2           ;R2 = BIT POINTER
10008  043222  010203                          MOV     R2,R3           ;R3 = BIT NUMBER
10009                                   ,MOVE R2 TO STARTING BIT OF ERROR BURST
10010
10011  043224  020301          30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
10012  043226  001403                          BEQ     35$             ;YES !!
10013  043230  006302                          ASL     R2              ;SHIFT BIT POINTER
10014  043232  005203                          INC     R3              ;INCREMENT BIT NUMBER
10015  043234  000773                          BR      30$
10016  043236  012703  000013          35$:    MOV     #11.,R3         ,R3 = LENGTH OF ERROR BURST
10017
10018                                   ,CORRECT THE ERROR BURST
```

```
10019  043242  030237  001376          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
10020  043246  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
10021  043250  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
10022  043252  001402                          BEQ     50$             ;NO
10023  043254  040210                          BIC     R2,(R0)         ;RESET THE BIT
10024  043256  000401                          BR      60$
10025  043260  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
10026  043262  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
10027  043264  001004                          BNE     70$
10028  043266  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT OF NEXT WORD
10029  043272  062700  000002                  ADD     #2,R0
10030  043276  005303                  70$:    DEC     R3              ;END OF BURST ??
10031  043300  001360                          BNE     40$             ;NO !!
10032  043302  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
10033  043306  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
10034  043314  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
10035  043320  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
10036  043326  013702  001332                  MOV     RMWC1,R2                ;R2 = NUMBER OF WORDS TRANSFER
10037  043332  163702  001402                  SUB     RMWC0,R2
10038  043336  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
10039  043340  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
10040  043342  005302                          DEC     R2                      ;DECREMENT WORD COUNT
10041  043344  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
10042  043346  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
10043  043350                          100$:
10044  043350  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
10045  043354  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
10046  043360  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
10047  043364  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
10048  043370  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
10049  043374  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
10050  043402  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
10051
10052                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
10053  043410  032737  100000  043450          BIT     #BIT15,150$     ;WAS ECC CORRECTION USED ??
10054  043416  001403                          BEQ     105$            ;NO !!
10055  043420  112776  000163  000010          MOVB    #163,@10(SP)    ;ECC CORRECTION FAILED
10056  043426  162766  000002  000010  105$:   SUB     #2,10(SP)               ;MOVE SP TO RETURN IF ERROR
10057  043434  000240                          NOP
10058  043436                          110$:
10059  043436  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10060  043440  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10061  043442  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10062  043444  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10063  043446  000207                          RTS     PC                      ;RETURN TO USER
10064
10065  043450  000000                  150$:   .WORD                   ;ECC CORRECTION FLAG
```

```
10066                                   .SBTTL  GET STATUS SUBROUTINE
10067                                   ;
10068                                   ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
10069                                   ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
10070                                   ;AND THEN RETURNS TO THE USER.
10071                                   ;
10072                                   ;CALL:  JSR     PC,GETSTS
10073                                   ;       ???                     RETURN HERE
10074
10075  043452                           GETSTS:
10076  043452  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10077  043454  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10078  043456  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10079  043460  012700  001506                  MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
10080  043464  012701  001400                  MOV     #RMEC21+2,R1    ;R1 = ADDRESS OF GET BUFFER
10081  043470  012702  000046                  MOV     #RMEC2,R2       ;R2 = REGISTER INDE
10082  043474  110220                  2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
10083  043476  005041                          CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
10084  043500  162702  000002          3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
10085  043504  100405                          BMI     4$              ;BRANCH OUT IF DONE
10086  043506  022702  000022                  CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
10087  043512  001370                          BNE     2$
10088  043514  005041                          CLR     -(R1)
10089  043516  000770                          BR      3$
10090  043520  112720  000200          4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
10091  043524  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10092  043526  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10093  043530  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10094  043532  000240                          NOP
10095  043534  000207                          RTS     PC
10096
```

```
10097                                   .SBTTL  GET SUBROUTINE
10098
10099                                   ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
10100                                   ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
10101                                   ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
10102                                   ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
10103                                   ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
10104                                   ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
10105                                   ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
10106                                   ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
10107                                   ;WHICH SHOULD FOLLOW THE LAST ENTRY.
10108                                   ;
10109                                   ;SUBROUTINE CALL:
10110                                   ;(1)    "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
10111                                   ;       VALUES AND TERMINATED WITH A CONTROL BYTE
10112                                   ;(2)    "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
10113                                   ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
10114                                   ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
10115                                   ;(3)    JSR     PC,GET
10116                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10117                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10118                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
10119                                   ;       ???
10120
10121  043536  000240                   GET:    NOP
10122  043540  062716  000004                   ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
10123  043544  105076  000000                   CLRB    @(SP)           ;ERROR CALL
10124  043550  162716  000004                   SUB     #4,(SP)
10125  043554  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10126  043556  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10127  043560  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10128  043562  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
10129  043564  010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
10130  043566  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10131  043572  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10132  043576  013700  001276                   MOV     $BASE,R0
10133  043602  012702  001330                   MOV     #GETBUF,R2
10134  043606  012704  001506                   MOV     #GETINX,R4
10135  043612  012737  043720  000004         MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
10136  043620  012737  000300  000006         MOV     #PR6,ERRVEC+2
10137  043626  016037  000010  001174  1$:    MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
10138  043634  016037  000000  001176         MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
10139  043642  032737  004000  001176         BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
10140  043650  001007                         BNE     3$              ;YES!!
10141  043652  062766  000004  000016         ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
10142  043660  112776  000112  000016         MOVB    #112,@16(SP)    ;ERROR CALL
10143  043666  000423                         BR      7$
10144  043670  105714                 3$:     TSTB    (R4)            ;DONE??
10145  043672  100433                         BMI     9$              ;YES!!
10146  043674  111401                         MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
10147  043676  042701  177700               BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
10148  043702  060001                         ADD     R0,R1
10149  043704  112403                         MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
10150  043706  042703  177700               BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
10151  043712  060203                         ADD     R2,R3
10152  043714  011113                         MOV     (R1),(R3)       ;READ REGISTER
```

```
10153  043716  000764                          BR      3$
10154
10155  043720  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
10156  043722  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10157  043730  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
10158  043736  162766  000002  000016  7$:     SUB     #2,16(SP)
10159  043744  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
10160  043746  100405                          BMI     9$              ;YES!!
10161  043750  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
10162  043752  112403                          MOVB    (R4)+,R3        ;LOCATIONS
10163  043754  060203                          ADD     R2,R3
10164  043756  005013                          CLR     (R3)
10165  043760  000771                          BR      8$
10166  043762                          9$:
10167  043762  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10168  043766  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10169  043772  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
10170  043774  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10171  043776  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10172  044000  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10173  044002  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10174  044004  000207                          RTS     PC              ;RETURN
10175
```

```
10176                                   .SBTTL  PUT SUBROUTINE
10177
10178                                   ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
10179                                   ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
10180                                   ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
10181                                   ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
10182                                   ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
10183                                   ;FOLLOW THE LAST ENTRY.
10184                                   ;
10185                                   ;SUBROUTINE CALL:
10186                                   ;
10187                                   ;(1)    "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
10188                                   ;       OF REGISTERS TO BE WRITTEN.
10189                                   ;(2)    "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
10190                                   ;       REGISTER TO BE WRITTEN.
10191                                   ;(3)    JSR     PC,PUT
10192                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10193                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10194                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
10195                                   ;       ???
10196
10197  044006  000240                   PUT:    NOP
10198  044010  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10199  044012  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10200  044014  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10201  044016  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
10202  044020  010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
10203  044022  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10204  044026  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10205  044032  013700  001276                   MOV     $BASE,R0
10206  044036  012702  001400                   MOV     #PUTBUF,R2
10207  044042  012704  001535                   MOV     #PUTINX,R4
10208  044046  012737  044154  000004           MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
10209  044054  012737  000300  000006           MOV     #PR6,ERRVEC+2
10210  044062  016037  000010  001174  1$:      MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
10211  044070  016037  000000  001176           MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
10212  044076  032737  004000  001176           BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
10213  044104  001007                           BNE     3$              ;YES!!
10214  044106  062766  000004  000016           ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10215  044114  112776  000112  000016           MOVB    #112,@16(SP)    ;USER'S ERROR CALL
10216  044122  000423                           BR      7$
10217  044124  105714                  3$:      TSTB    (R4)            ;DONE??
10218  044126  100424                           BMI     9$              ;YES!!
10219  044130  111401                           MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
10220  044132  042701  177700                   BIC     #^C<IDXMSK>,R1  ;CLEAR ANY SIGN EXTENSION
10221  044136  060001                           ADD     R0,R1
10222  044140  112403                           MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
10223  044142  042703  177700                   BIC     #^C<IDXMSK>,R3  ;CLEAR ANY SIGN EXTENSION
10224  044146  060203                           ADD     R2,R3
10225  044150  011311                           MOV     (R3),(R1)       ;WRITE REGISTER
10226  044152  000764                           BR      3$
10227
10228  044154  022626                  5$:      CMP     (SP)+,(SP)+     ;ADJUST STACK
10229  044156  062766  000004  000016           ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10230  044164  112776  000007  000016           MOVB    #7,@16(SP)      ;USER'S ERROR CALL
10231  044172  162766  000002  000016  7$:      SUB     #2,16(SP)
```

```
10232
10233  044200                          9$:
10234  044200  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10235  044204  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10236  044210  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
10237  044212  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10238  044214  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10239  044216  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10240  044220  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10241  044222  000207                          RTS     PC              ;RETURN
10242
```

```
10243                                   .SBTTL  SIZE CLOCK SUBROUTINE
10244
10245  044224                           SIZCLK:
10246  044224  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10247  044230  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10248  044234  012737  044272  000004           MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10249  044242  012737  000300  000006           MOV     #PR6,ERRVEC+2
10250  044250  012737  177546  001502           MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
10251  044256  012737  000100  001504           MOV     #100,CLKVCT
10252  044264  005777  135212                   TST     @CLKADR         ;TEST FOR KW11-L PRESENT
10253  044270  000421                           BR      3$              ;YES - KW11-L IS PRESENT
10254  044272  022626                   1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
10255  044274  012737  044324  000004           MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10256  044302  012737  172540  001502           MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
10257  044310  012737  000104  001504           MOV     #104,CLKVCT
10258  044316  005777  135160                   TST     @CLKADR         ;TEST FOR KW11-P PRESENT
10259  044322  000404                           BR      3$              ;YES - KW11-P IS PRESENT
10260  044324  022626                   2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
10261  044326  062766  000002  000004           ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
10262  044334                           3$:
10263  044334  012637  000006                   MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10264  044340  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10265  044344  000207                           RTS     PC              ;RETURN TO USER
```

```
10266                                   .SBTTL  TIMEOUT SUBROUTINE
10267                                   ;
10268                                   ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
10269                                   ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
10270                                   ;
10271                                   ;CALL:  JSR     PC,TIMOUT
10272                                   ;       ???                     RETURN HERE
10273
10274  044346                           TIMOUT:
10275  044346  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10276  044350  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10277  044352  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10278  044354  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10279  044360  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10280  044364  012737  044466  000004          MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
10281  044372  012737  000300  000006          MOV     #PR6,ERRVEC+2
10282  044400  013700  001276                  MOV     $BASE,R0        ;R0=RM03 ADDRESS
10283  044404  013701  001502                  MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
10284  044410  012702  000037                  MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
10285  044414  020127  172540          1$:     CMP     R1,#172540      ;KW11-P CLOCK??
10286  044420  001003                          BNE     2$              ;NO!!
10287  044422  012761  000001  000002          MOV     #1,2(R1)        ;SET COUNTER
10288  044430  012711  000005          2$:     MOV     #BIT2!BIT0,(R1) ;START COUNTER
10289
10290  044434  016046  000000          3$:     MOV     RMCS1(R0),-(SP) ;GET STATUS
10291  044440  042716  177576                  BIC     #^C<RDY!GO>,(SP)
10292  044444  022726  000200                  CMP     #RDY,(SP)+      ;RDY=1,GO=0??
10293  044450  001420                          BEQ     5$              ;YES!!
10294  044452  032711  000200                  BIT     #BIT7,(R1)      ;TIMER DONE??
10295  044456  001766                          BEQ     3$              ;NO!!
10296  044460  005302                          DEC     R2              ;DEC NUMBER OF CYCLES
10297  044462  001354                          BNE     1$              ;CONTINUE IF NOT DONE
10298  044464  000412                          BR      5$
10299  044466  022626                  4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
10300  044470  062766  000004  000012          ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
10301  044476  112776  000007  000012          MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
10302  044504  162766  000002  000012          SUB     #2,12(SP)
10303
10304  044512                          5$:
10305  044512  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10306  044516  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10307  044522  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10308  044524  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10309  044526  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10310  044530  000207                          RTS     PC              ;RETURN TO USER
10311
```

```
10312                                   .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
10313                                   ;
10314                                   ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
10315                                   ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
10316                                   ;FOLLOWING CHECKS ARE MADE:
10317                                   ;
10318                                   ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
10319                                   ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
10320                                   ;
10321                                   ;       .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
10322                                   ;AND NED (BIT 12 OF RMCS2) IS RESET;
10323                                   ;
10324                                   ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
10325                                   ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
10326                                   ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
10327                                   ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
10328                                   ;I.E., MCPE = 0.
10329                                   ;       .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
10330                                   ;I.E., PAR = 0, OR, PAR = DPE = 1
10331                                   ;
10332                                   ;
10333                                   ;THE SUBROUTINE ASSUMES THAT:
10334                                   ;
10335                                   ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
10336                                   ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
10337                                   ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
10338                                   ;
10339                                   ;       .($UNIT) CONTAINS THE DRIVE NUMBER
10340                                   ;
10341                                   ;THE SUBROUTINE IS CALLED AS FOLLOWS:
10342                                   ;
10343                                   ;(1)    JSR     PC,PRIERR
10344                                   ;       BR      ???             RETURN HERE IF NO ERROR
10345                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10346                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10347                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10348                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10349
10350  044532                           PRIERR:
10351
10352                                   ;CLEAR USER'S ERROR CALL
10353  044532  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10354  044536  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
10355  044542  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10356
10357
10358                                   ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
10359  044546  013737  001340  001142          MOV     RMCS21,$BDDAT   ;CORRECT UNIT SELECTED??
10360  044554  042737  177770  001142          BIC     #^CUNTMSK,$BDDAT
10361  044562  013737  001234  001140          MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
10362  044570  042737  177770  001140          BIC     #^CUNTMSK,$GDDAT
10363  044576  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10364                                                                  ;DRIVE NUMBERS
10365  044604  001415                          BEQ     1$              ;YES!!
10366  044606  062716  000004                  ADD     #4,(SP)
10367  044612  112776  000001  000000          MOVB    #1,@(SP)        ;ERROR 1
```

```
10368  044620  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10369  044624  004736                    JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
10370  044626  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
10371  044632  000240                    NOP
10372  044634  000137  045354            JMP     10$              ;SKIP OTHER CHECKS
10373  044640                          1$:
10374
10375                                   ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
10376                                   ;THE DEVICE IS NONEXISTANT
10377  044640  032737  004000  001330            BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
10378  044646  001045                    BNE     5$               ;YES!!
10379  044650  013737  001330  001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10380  044656  052737  004000  001140    BIS     #DVA,$GDDAT
10381  044664  013737  001330  001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10382  044672  062716  000004            ADD     #4,(SP)
10383  044676  112776  000002  000000    MOVB    #2,@(SP)         ;ERROR #2
10384  044704  032737  010000  001340    BIT     #NED,RMCS2I      ;WAS NED SET??
10385  044712  001414                    BEQ     2$               ;NO!!
10386  044714  013737  001340  001140    MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
10387  044722  013737  001340  001142    MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
10388  044730  042737  010000  001140    BIC     #NED,$GDDAT
10389  044736  112776  000003  000000    MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
10390  044744  162716  000002          2$:SUB    #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10391  044750  004736                    JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
10392  044752  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
10393  044756  000240                    NOP
10394  044760  000575                    BR      10$              ;SKIP OTHER CHECKS
10395  044762                          5$:
10396
10397                                   ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
10398  044762  032737  000200  001330            BIT     #RDY,RMCS1I      ;CONTROLLER READY??
10399  044770  001030                    BNE     7$               ;YES!!
10400  044772  013737  001330  001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10401  045000  052737  000200  001140    BIS     #RDY,$GDDAT
10402  045006  042737  160001  001140    BIC     #SC!TRE!MCPE!GO,$GDDAT
10403  045014  013737  001330  001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10404  045022  062716  000004            ADD     #4,(SP)
10405  045026  112776  000004  000000    MOVB    #4,@(SP)         ;ERROR #4
10406  045034  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10407  045040  004736                    JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
10408  045042  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
10409  045046  000240                    NOP
10410  045050  000541                    BR      10$              ;SKIP OTHER CHECKS
10411  045052                          7$:
10412
10413                                   ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
10414  045052  032737  000001  001330            BIT     #GO,RMCS1I       ;GO RESET??
10415  045060  001431                    BEQ     8$               ;YES!!
10416  045062  032737  000200  001342            BIT     #DRY,RMDS1       ;DRIVE READY??
10417  045070  001025                    BNE     8$               ;YES!!
10418  045072  013737  001330  001140    MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10419  045100  042737  160001  001140    BIC     #SC!TRE!MCPE!GO,$GDDAT
10420  045106  013737  001330  001142    MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10421  045114  062716  000004            ADD     #4,(SP)
10422  045120  112776  000005  000000    MOVB    #5,@(SP)         ;ERROR #5
10423  045126  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
10424  045132  004736                          JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
10425  045134  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10426  045140  000240                          NOP
10427  045142  000504                          BR      10$
10428  045144                          8$:
10429
10430                                          ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
10431                                          ;PARITY ON THE MASSBUS CONTROL BUS
10432  045144  032737  020000  001330          BIT     #MCPE,RMCS11    ;PARITY ERROR ??
10433  045152  001425                          BEQ     9$              ;NO!!
10434  045154  013737  001330  001140          MOV     RMCS11,$GDDAT   ;EXPECTED STATUS
10435  045162  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
10436  045170  013737  001330  001142          MOV     RMCS11,$BDDAT   ;RECEIVED STATUS
10437  045176  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
10438  045202  112776  000013  000000          MOVB    #13,@(SP)       ;ERROR #47
10439  045210  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10440  045214  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10441  045216  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
10442  045222  000240                          NOP
10443  045224  000453                          BR      10$
10444  045226                          9$:
10445
10446                                          ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
10447  045226  032737  000010  001344          BIT     #PAR,RMER11     ;WAS THERE A PARITY ERROR??
10448  045234  001451                          BEQ     11$             ;NO!!
10449  045236  032737  000010  001372          BIT     #DPE,RMER21     ;WAS IT THE CONTROL BUS??
10450  045244  001045                          BNE     11$             ;NOT SURE!!
10451  045246  032737  000010  001414          BIT     #PAR,RMER10     ;DID TEST SET PAR ??
10452  045254  001413                          BEQ     93$             ;NO!!
10453  045256  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10454  045260  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
10455  045264  122710  000014          91$:    CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
10456  045270  001002                          BNE     92$
10457  045272  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10458  045274  000431                          BR      11$             ;PAR WAS SET BY TEST
10459  045276  105720                  92$:    TSTB    (R0)+           ;END OF TABLE??
10460  045300  100371                          BPL     91$             ;NO!!
10461  045302  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10462  045304  013737  001344  001140  93$:    MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10463  045312  042737  000010  001140          BIC     #PAR,$GDDAT
10464  045320  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10465  045326  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10466  045332  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
10467  045340  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10468  045344  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
10469  045346  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10470  045352  000240                          NOP
10471  045354  062716  000010          10$:    ADD     #10,(SP)                ;RETURN TO ERROR
10472  045360  000240                  11$:    NOP                     ;RETURN TO NO ERROR
10473  045362  000207                          RTS     PC
10474
```

```
10475                                   .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
10476                                   ;
10477                                   ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
10478                                   ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
10479                                   ;CONTENTS.  THESE ERRORS ARE DEEMED
10480                                   ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
10481                                   ;BEING PERFORMED.  WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
10482                                   ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
10483                                   ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL.  AFTER THE TEST ROUTINE
10484                                   ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
10485                                   ;OTHER ERRORS.  WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
10486                                   ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
10487                                   ;
10488                                   ;CALL:  JSR     PC,SECERR
10489                                   ;       BR      ???             RETURN HERE IF NO ERROR
10490                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10491                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10492                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10493                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10494
10495                                   ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
10496                                   ;INPUT REGISTER BUFFER.
10497
10498  045364                           SECERR:
10499
10500                                   ;;*****************************************************************
10501                                   ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
10502  045364  013737  001400  051224           MOV     RMCS10,$15$     ;STORE FUNCTION CODE
10503  045372  042737  177701  051224           BIC     #^C<F0!F1!F2!F3!F4>,$15$
10504  045400  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10505  045404  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
10506  045410  162716  000004                   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10507
10508                                   ;;*****************************************************************
10509                                   ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
10510
10511                                   ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
10512  045414  032737  000200  001342           BIT     #DRY,RMDS1      ;DRIVE READY??
10513  045422  001024                           BNE     5$              ;YES!!
10514  045424  013737  001342  001142           MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
10515  045432  042737  177577  001142           BIC     #^CDRY,$BDDAT
10516  045440  012737  000200  001140           MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
10517  045446  062716  000004                   ADD     #4,(SP)
10518  045452  112776  000010  000000           MOVB    #10,@(SP)       ;ERROR NUMBER
10519  045460  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10520  045464  004736                           JSR     PC,@(SP)+       ;REPORT NOT READY
10521  045466  162716  000010                   SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
10522  045472  000240                           NOP
10523
10524                                   ;REPORT ERROR IF GO BIT IS NOT RESET
10525  045474  032737  000001  001330  5$:      BIT     #GO,RMCS11      ;GO BIT RESET??
10526  045502  001423                           BEQ     10$             ;YES!.
10527  045504  013737  001330  001142           MOV     RMCS11,$BDDAT   ;BAD DATA FOR TYPEOUT
10528  045512  042737  177776  001142           BIC     #^CGO,$BDDAT
10529  045520  005037  001140                   CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
10530  045524  062716  000004                   ADD     #4,(SP)
```

```
10531  045530  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
10532  045536  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10533  045542  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
10534  045544  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10535  045550  000240                          NOP
10536
10537                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
10538  045552  013737  001330  001142  10$:    MOV     RMCS11,$BDDAT   ;IS FUNCTION CODE CORRECT??
10539  045560  042737  177701  001142          BIC     #^C76,$BDDAT
10540  045566  013737  051224  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
10541  045574  023737  001142  001140          CMP     $BDDAT,$GDDAT
10542  045602  001413                          BEQ     15$             ;YES!!
10543  045604  062716  000004                  ADD     #4,(SP)
10544  045610  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
10545  045616  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10546  045622  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
10547  045624  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10548  045630  000240                          NOP
10549  045632                          15$:
10550                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
10551                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
10552                                  ;OTHER ERRORS ARE SET
10553  045632  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
10554  045636  005737  001344                  TST     RMER1           ;IS RMER1 -0??
10555  045642  001003                          BNE     20$             ;NO!!
10556  045644  005737  001372                  TST     RMER21          ;IS RMER2=0??
10557  045650  001403                          BEQ     25$             ;YES!!
10558  045652  052737  040000  001140  20$:    BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
10559  045660  013737  001342  001142  25$:    MOV     RMDS1,$BDDAT
10560  045666  042737  137777  001142          BIC     #^CERR,$BDDAT
10561  045674  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
10562  045702  001412                          BEQ     30$             ;YES!!
10563  045704  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10564  045710  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
10565  045716  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10566  045722  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
10567  045724  162716  000010                  SUB     #10,(SP)
10568
10569                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
10570                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
10571                                   SET TRE IS SET
10572  045730  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
10573  045734  013746  001340                  MOV     RMCS21,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
10574  045740  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
10575  045744  001010                          BNE     35$             ;YES!!
10576  045746  032737  040000  001342          BIT     #ERR,RMDS1      ;WAS EXCEPTION RECEIVED??
10577  045754  001407                          BEQ     40$             ;NO!!
10578  045756  022737  000030  051224          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10579  045764  103003                          BHIS    40$             ;NO!!
10580  045766  052737  040000  001140  35$:    BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
10581  045774  013737  001330  001142  40$:    MOV     RMCS11,$BDDAT   ;BAD DATA FOR TYPEOUT
10582  046002  042737  137777  001142          BIC     #^CTRE,$BDDAT
10583  046010  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
10584  046016  001413                          BEQ     45$             ;YES!!
10585  046020  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10586  046024  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
10587  046032  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10588  046036  004736                          JSR     PC,@(SP)+        ;REPORT TRE ERROR
10589  046040  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
10590  046044  000240                          NOP
10591  046046                          45$:
10592
10593                                  ;;*****************************************************************
10594                                  ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
10595                                  ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
10596                                  ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
10597                                  ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
10598                                  ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
10599                                  ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
10600
10601                                  ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
10602  046046  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
10603  046050  013700  051224                  MOV     51$,R0           ;R0 = FUNCTION CODE
10604  046054  016037  070674  051216          MOV     FNCDIB(R0),500$  ;STORE ENTRY
10605  046062  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
10606
10607                                  ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
10608                                  ;ATA IS NOT SET AND SHOULD BE SET.
10609  046064  013737  051216  001140          MOV     500$,$GDDAT      ;GET EXPECTED ATA STATUS
10610  046072  032737  040000  001342          BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
10611  046100  001403                          BEQ     50$              ;NO !!
10612  046102  052737  100000  001140          BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
10613  046110  042737  077777  001140  50$:     BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
10614  046116  013737  001342  001142          MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
10615  046124  042737  077777  001142          BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
10616  046132  023737  001140  001142          CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
10617  046140  001413                          BEQ     55$              ;YES !!
10618  046142  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10619  046146  112776  000006  000000          MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
10620  046154  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10621  046160  004736                          JSR     PC,@(SP)+        ;REPORT ERROR
10622  046162  162716  000010                  SUB     #10,(SP)         ;RESTORE SP
10623  046166  000240                          NOP
10624  046170                          55$:
10625
10626                                  ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
10627                                  ;WITH FUNCTION CODE TABLE
10628  046170  013737  051216  001140          MOV     500$,$GDDAT      ;GET EXPECTED ILF
10629  046176  042737  177776  001140          BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
10630  046204  013737  001344  001142          MOV     RMER11,$BDDAT    ;GET RECEIVED ILF
10631  046212  042737  177776  001142          BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
10632  046220  023737  001140  001142          CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
10633  046226  001412                          BEQ     60$              ;YES !!
10634  046230  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10635  046234  112776  000254  000000          MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
10636  046242  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10637  046246  004736                          JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
10638  046250  162716  000010                  SUB     #10,(SP)         ;MOVE SP TO NO ERROR
10639  046254  005037  001140          60$:    CLR     $GDDAT           ;CLEAR EXPECTED STATUS
10640
10641                                  ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
10642  046260  013746  051216                  MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
10643  046264  052716  137777                 BIS     #^CWCE,(SP)     ;SET ALL OTHER BITS
10644  046270  013737  001340  001142         MOV     RMCS21,$BDDAT   ;RECEIVED STATUS
10645  046276  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
10646  046302  001412                         BEQ     90$             ;BRANCH IF WCE OK
10647  046304  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10648  046310  112776  000026  000000         MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
10649  046316  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10650  046322  004736                         JSR     PC,@(SP)+       ;REPORT ERROR
10651  046324  162716  000010                 SUB     #10,(SP)        ;RESTORE ERROR
10652  046330                         90$:
10653
10654                                 ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
10655  046330  013746  051216                 MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
10656  046334  052716  157777                 BIS     #^COPI,(SP)     ;SET ALL OTHER BITS
10657  046340  013737  001344  001142         MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
10658  046346  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
10659  046352  001412                         BEQ     100$            ;BRANCH IF OPI OK
10660  046354  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10661  046360  112776  000164  000000         MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
10662  046366  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10663  046372  004736                         JSR     PC,@(SP)+       ;REPORT ERROR
10664  046374  162716  000010                 SUB     #10,(SP)        ;RESTORE SP
10665  046400                         100$:
10666
10667                                 ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
10668                                 ;SET AND VV IS NOT RESET
10669  046400  013746  051216                 MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
10670  046404  032737  000100  001342         BIT     #VV,RMDS1       ;IS VV SET
10671  046412  001402                         BEQ     105$            ;NO !!
10672  046414  042716  010000                 BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
10673  046420  052716  167777         105$:   BIS     #^CIVC,(SP)     ;SET ALL OTHER BITS
10674  046424  013737  001372  001142         MOV     RMER21,$BDDAT   ;GET RECEIVED STATUS
10675  046432  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
10676  046436  001412                         BEQ     110$            ;BRANCH IF IVC OK
10677  046440  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10678  046444  112776  000165  000000         MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
10679  046452  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10680  046456  004736                         JSR     PC,@(SP)+       ;REPORT ERROR
10681  046460  162716  000010                 SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10682  046464                         110$:
10683
10684                                 ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10685                                 ; ALL WRITE ERRORS, I.E.,
10686                                 ;       RMER1 - WLE, WCF
10687                                 ;       RMER2 - DPE
10688                                 ;       RMCS2 - UPE.
10689                                 ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
10690                                 ;WRITE ERROR ENABLE BIT IS RESET.
10691
10692                                 ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
10693                                 ;THE DRIVE IS NOT WRITE PROTECTED
10694  046464  012746  177777                 MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10695  046470  032737  004000  051216         BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10696  046476  001404                         BEQ     115$            ;NO !!
10697  046500  032737  004000  001342         BIT     #WRL,RMDS1      ;IS THE DRIVE WRITE PROTECTED ??
10698  046506  001002                         BNE     120$            ;YES !!
```

```
10699  046510  042716  004000          115$:   BIC     #WLE,(SP)       ;RESET WLE ENABLE
10700  046514  013737  001344  001142  120$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10701  046522  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WLE IF ENABLED
10702  046526  001412                          BEQ     125$            ;BRANCH IF WLE OK
10703  046530  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10704  046534  112776  000023  000000          MOVB    #23,@(SP)       ;WRITE ERROR NUMBER IN CALL
10705  046542  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10706  046546  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10707  046550  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10708  046554                          125$:
10709
10710                                  ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
10711  046554  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10712  046560  032737  004000  051216          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10713  046566  001002                          BNE     130$            ;YES !!
10714  046570  042716  000040                  BIC     #WCF,(SP)       ;DISABLE WCF ERROR
10715  046574  013737  001344  001142  130$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10716  046602  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET WCF IF ENABLED
10717  046606  001412                          BEQ     135$            ;BRANCH IF WCF OK
10718  046610  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10719  046614  112776  000025  000000          MOVB    #25,@(SP)       ;WRITE ERROR NUMBER IN CALL
10720  046622  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10721  046626  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10722  046630  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10723  046634                          135$:
10724
10725                                  ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10726  046634  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
10727  046640  032737  004000  051216          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10728  046646  001002                          BNE     140$            ;YES !!
10729  046650  042716  000010                  BIC     #DPE,(SP)       ;RESET DPE ENABLE
10730  046654  013737  001372  001142  140$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10731  046662  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET DPE IF ENABLED
10732  046666  001412                          BEQ     145$            ;BRANCH IF DPE OK
10733  046670  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10734  046674  112776  000040  000000          MOVB    #40,@(SP)       ;WRITE ERROR NUMBER IN CALL
10735  046702  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10736  046706  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10737  046710  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10738  046714                          145$:
10739
10740                                  ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10741  046714  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
10742  046720  032737  004000  051216          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10743  046726  001002                          BNE     150$            ;YES !!
10744  046730  042716  020000                  BIC     #UPE,(SP)       ;DISABLE UPE ERROR
10745  046734  013737  001340  001142  150$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10746  046742  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET UPE IF ENABLED
10747  046746  001412                          BEQ     155$            ;BRANCH IF UPE OK
10748  046750  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10749  046754  112776  000024  000000          MOVB    #24,@(SP)       ;WRITE ERROR NUMBER IN CALL
10750  046762  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10751  046766  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10752  046770  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10753  046774                          155$:
10754
```

```
10755                                   ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
10756  046774  013746  051216                    MOV     500$,-(SP)      ;GET IAE ENABLE
10757  047000  052716  175777                    BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
10758  047004  013737  001344  001142            MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10759  047012  042637  001142                    BIC     (SP)+, $BDDAT   ;CLEAR IAE IF ENABLED
10760  047016  001412                            BEQ     160$            ;BRANCH IF IAE IS OK
10761  047020  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10762  047024  112776  000166  000000            MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
10763  047032  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10764  047036  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10765  047040  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10766  047044                          160$:
10767
10768                                   ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10769                                   ; ALL READ/WRITE ERRORS, I.E.,
10770                                   ;       RMCS1 - TRE
10771                                   ;       RMCS2 - DLT,NEM,MXF
10772                                   ;       RMDS  - LBT
10773                                   ;       RMER1 - AOE
10774                                   ;NOTE:
10775                                   ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
10776                                   ;CYLINDER REGISTER IS WRITTEN
10777                                   ;NOTE:
10778                                   ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
10779                                   ;NOTE:
10780                                   ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
10781
10782
10783                                   ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10784  047044  012746  177777                    MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10785  047050  032737  001000  051216            BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10786  047056  001002                            BNE     165$            ;YES !!
10787  047060  042716  100000                    BIC     #DLT,(SP)       ;RESET DLT ENABLE
10788  047064  013737  001340  001142  165$:     MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10789  047072  042637  001142                    BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
10790  047076  001412                            BEQ     170$            ;BRANCH IF DLT IS OK
10791  047100  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10792  047104  112776  000032  000000            MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
10793  047112  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10794  047116  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10795  047120  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10796  047124                          170$:
10797
10798                                   ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10799  047124  012746  177777                    MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10800  047130  032737  001000  051216            BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10801  047136  001002                            BNE     175$            ;YES !!
10802  047140  042716  004000                    BIC     #NEM,(SP)       ;DISABLE NEM
10803  047144  013737  001340  001142  175$:     MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10804  047152  042637  001142                    BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
10805  047156  0014'2                            BEQ     180$            ;BRANCH IF NEM IS OK
10806  047160  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10807  047164  112776  000167  000000            MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
10808  047172  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10809  047176  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10810  047200  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
10811  047204                                  180$:
10812
10813                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10814  047204  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10815  047210  032737  001000  051216                BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
10816  047216  001002                                BNE     185$            ;YES !!
10817  047220  042716  001000                        BIC     #MXF,(SP)       ;DISABLE MXF ERROR
10818  047224  013737  001340  001142  185$:         MOV     RMCS21,$BDDAT   ;GET RECEIVED STATUS
10819  047232  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
10820  047236  001412                                BEQ     190$            ;BRANCH IF MXF IS OK
10821  047240  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10822  047244  112776  000033  000000                MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
10823  047252  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10824  047256  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10825  047260  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10826  047264                                  190$:
10827
10828                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
10829  047264  012746  177777                        MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
10830  047270  032737  001000  051216                BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
10831  047276  001404                                BEQ     191$            ;NO !!
10832  047300  032737  002000  001342                BIT     #LBT,RMDS1      ;IS LBT ALSO SET ??
10833  047306  001002                                BNE     195$            ;YES !!
10834  047310  042716  001000          191$:         BIC     #AOE,(SP)       ;DISABLE AOE
10835  047314  013737  001344  001142  195$:         MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
10836  047322  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
10837  047326  001412                                BEQ     200$            ;BRANCH IF AOE IS OK
10838  047330  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10839  047334  112776  000020  000000                MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
10840  047342  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10841  047346  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10842  047350  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10843  047354                                  200$:
10844
10845                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10846                                          ;HEADER ERRORS, I.E.,
10847                                          ;       RMER1 - HCRC,HCE,FER
10848                                          ;       RMER2 - BSE
10849
10850                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
10851  047354  032737  002000  001362                BIT     #HCI,RMOF1      ;IS HCI SET ??
10852  047362  001403                                BEQ     201$            ;NO !!
10853  047364  042737  000200  051216                BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
10854  047372                                  201$:
10855
10856                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
10857  047372  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10858  047376  032737  000200  051216                BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10859  047404  001002                                BNE     205$            ;YES !!
10860  047406  042716  000400                        BIC     #HCRC,(SP)      ;DISABLE HCRC
10861  047412  013737  001344  001142  205$:         MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
10862  047420  042637  001142                        BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
10863  047424  001412                                BEQ     210$            ;BRANCH IF HCRC IS OK
10864  047426  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10865  047432  112776  000035  000000                MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
10866  047440  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
10867  047444  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10868  047446  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10869  047452                        210$:
10870
10871                                ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
10872  047452  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10873  047456  032737  000200  051216        BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
10874  047464  001002                        BNE     215$            ;YES !!
10875  047466  042716  000200                BIC     #HCE,(SP)       ;DISABLE HCE
10876  047472  013737  001344  001142 215$:  MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
10877  047500  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
10878  047504  001412                        BEQ     220$            ;BRANCH IF HCE IS OK
10879  047506  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10880  047512  112776  000036  000000        MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
10881  047520  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10882  047524  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10883  047526  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10884  047532                        220$:
10885
10886                                ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
10887  047532  012746  177777                MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
10888  047536  032737  000200  051216        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10889  047544  001002                        BNE     225$            ;YES !!
10890  047546  042716  000020                BIC     #FER,(SP)       ;DISABLE FER
10891  047552  013737  001344  001142 225$:  MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
10892  047560  042637  001142                BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
10893  047564  001412                        BEQ     230$            ;BRANCH IF FER OK
10894  047566  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10895  047572  112776  000037  000000        MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
10896  047600  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10897  047604  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10898  047606  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10899  047612                        230$:
10900
10901                                ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
10902  047612  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10903  047616  032737  000200  051216        BIT     #HCE,500$       ;ARE THEY ENABLED ??
10904  047624  001002                        BNE     235$            ;YES !!
10905  047626  042716  100000                BIC     #BSE,(SP)       ;DISABLE BSE
10906  047632  013737  001372  001142 235$:  MOV     RMER21,$BDDAT   ;GET RECEIVED STATUS
10907  047640  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
10908  047644  001412                        BEQ     240$            ;BRANCH IF BSE OK
10909  047646  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10910  047652  112776  000354  000000        MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
10911  047660  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10912  047664  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10913  047666  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10914  047672                        240$:
10915
10916                                ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
10917                                ;FIELD ERRORS, I.E.,
10918                                ;       RMCS2 - MDPE
10919                                ;       RMER1 - DCK,ECH
10920                                ;NOTE:
10921                                ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
10922                                ;DCK IS SET.
```

```
10923
10924                                           ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
10925  047672  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10926  047676  032737  000100  051216                  BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
10927  047704  001002                                  BNE     245$            ;YES !!
10928  047706  042716  000400                          BIC     #MDPE,(SP)      ;DISBALE MDPE
10929  047712  013737  001340  001142  245$:           MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10930  047720  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
10931  047724  001412                                  BEQ     250$            ;BRANCH IF MDPE OK
10932  047726  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10933  047732  112776  000027  000000                  MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
10934  047740  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10935  047744  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10936  047746  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10937  047752                          250$:
10938
10939                                           ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
10940  047752  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10941  047756  032737  000100  051216                  BIT     #ECH,500$       ;ARE THEY ENABLED ??
10942  047764  001002                                  BNE     255$            ;YES !!
10943  047766  042716  100000                          BIC     #DCK,(SP)       ;DISABLE DCK
10944  047772  013737  001344  001142  255$:           MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10945  050000  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
10946  050004  001412                                  BEQ     260$            ;BRANCH IF DCK IS OK
10947  050006  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10948  050012  112776  000030  000000                  MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
10949  050020  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10950  050024  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10951  050026  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10952  050032                          260$:
10953
10954                                           ;REPORT ERROR IF ECH IS SET AND,
10955                                           ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
10956                                           ;       ECI IS SET, OR
10957                                           ;       DCK IS NOT SET.
10958  050032  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10959  050036  032737  000100  051216                  BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
10960  050044  001410                                  BEQ     265$            ;NO !!
10961  050046  032737  004000  001362                  BIT     #ECI,RMOFI      ;IS ECI SET ??
10962  050054  001004                                  BNE     265$            ;YES !!
10963  050056  032737  100000  001344                  BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
10964  050064  001002                                  BNE     270$            ;YES !!
10965  050066  042716  000100          265$:           BIC     #ECH,(SP)       ;DISABLE ECH
10966  050072  013737  001344  001142  270$:           MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10967  050100  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
10968  050104  001412                                  BEQ     275$            ;BRANCH IF ECH IS OK
10969  050106  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10970  050112  112776  000031  000000                  MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
10971  050120  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10972  050124  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10973  050126  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10974  050132                          275$:
10975
10976                                           ;;*****************************************************************
10977                                           ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
10978
```

```
10979  050132  022737  000030  051224          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10980  050140  103402                          BLO     280$            ;YES!!
10981  050142  000137  051170                  JMP     355$
10982
```

```
10983                                   ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
10984                                   ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10985  050146  013737  001332  001142  280$:   MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
10986  050154  001421                          BEQ     285$            ;YES
10987  050156  032737  040000  001330          BIT     #TRE,RMCS11     ;TRANSFER ERROR DETECTED??
10988  050164  001015                          BNE     285$            ;YES!!
10989  050166  062716  000004                  ADD     #4,(SP)
10990  050172  112776  000015  000000          MOVB    #15,@(SP)       ;ERROR NUMBER
10991  050200  005037  001140                  CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
10992  050204  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10993  050210  004736                          JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
10994  050212  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10995  050216  000240                          NOP
10996                                   ;REPORT ERROR IF RMBA IS NOT CORRECT
10997  050220  013737  001332  001140  285$:   MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
10998  050226  163737  001402  001140          SUB     RMWCO,$GDDAT
10999  050234  006337  001140                  ASL     $GDDAT
11000  050240  063737  001404  001140          ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
11001  050246  032737  000010  001340          BIT     #BAI,RMCS2I     ;WAS BAI SET ??
11002  050254  001403                          BEQ     290$            ;NO !!
11003  050256  013737  001404  001140          MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
11004  050264  023737  001140  001334  290$:   CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
11005  050272  001416                          BEQ     295$            ;YES!!
11006  050274  013737  001334  001142          MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
11007  050302  062716  000004                  ADD     #4,(SP)
11008  050306  112776  000016  000000          MOVB    #16,@(SP)       ;ERROR NUMBER
11009  050314  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11010  050320  004736                          JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
11011  050322  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11012  050326  000240                          NOP
11013                                   ;COMPUTE NUMBER OF SECTORS TRANSFERRED
11014  050330  005046                  295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
11015  050332  013746  001332                  MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
11016  050336  163716  001402                  SUB     RMWCO,(SP)
11017  050342  012746  000400                  MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
11018  050346  032737  000002  001400          BIT     #BIT1,RMCS10    ;HEADER & DATA COMMAND ??
11019  050354  001402                          BEQ     300$            ;NO !!
11020  050356  012716  000402                  MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
11021  050362  005266  000004          300$:   INC     4(SP)           ;INCREMENT SECTOR COUNT
11022  050366  161666  000002                  SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
11023  050372  003373                          BGT     300$            ;CONTINUE IF NOT DONE
11024  050374  022626                          CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
11025                                   ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
11026  050376  013737  001406  051222          MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
11027  050404  013737  001406  051220          MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
11028  050412  013737  001434  051216          MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
11029  050420  042737  177740  051222          BIC     #^C37,510$
11030  050426  000337  051220                  SWAB    505$
11031  050432  042737  177770  051220          BIC     #^C7,505$
11032  050440  062637  051222                  ADD     (SP)+,510$
11033
11034  050444  023727  051222  000040  305$:   CMP     510$,#32.       ;SECTOR OVEFLOWED??
11035  050452  103420                          BLO     315$            ;NO!!
11036  050454  023727  051222  000240          CMP     510$,#<5*32.>   ;CYLINDERS WORTH??
11037  050462  103406                          BLO     310$            ;NO!!
11038  050464  005237  051216                  INC     500$            ;YES INCREMENT CYLINDER
```

```
11039  050470  162737  000240  051222          SUB     #<5*32.>,510$   ;ADJUST SECTOR
11040  050476  000762                          BR      305$            ;TRY AGAIN
11041  050500  005237  051220          310$:   INC     505$            ;INCREMENT TRACK
11042  050504  162737  000040  051222          SUB     #32.,510$       ;ADJUST SECTOR
11043  050512  000754                          BR      305$            ;TRY AGAIN
11044
11045  050514  023727  051220  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
11046  050522  103406                          BLO     320$            ;NO!!
11047  050524  005237  051216                  INC     500$            ;INCREMENT CYLINDER
11048  050530  162737  000005  051220          SUB     #5,505$ ;ADJUST TRACK
11049  050536  000766                          BR      315$            ;TRY AGAIN
11050                                  ;REPORT ERROR IF "LBT" IS NOT CORRECT
11051  050540  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
11052  050544  022737  001467  051216          CMP     #823.,500$      ;SHOULD LBT BE SET??
11053  050552  101007                          BHI     325$            ;NO!!
11054  050554  032737  002000  001344          BIT     #IAE,RMER11     ;WAS IAE SET ??
11055  050562  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
11056  050564  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
11057  050572  013737  001342  001142  325$:   MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
11058  050600  042737  175777  001142          BIC     #^CLBT,$BDDAT
11059  050606  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
11060  050614  001413                          BEQ     330$            ;YES!!
11061  050616  062716  000004                  ADD     #4,(SP)
11062  050622  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
11063  050630  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11064  050634  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
11065  050636  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11066  050642  000240                          NOP
11067                                  ;REPORT ERROR IF "AOE" IS INCORRECT
11068  050644  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
11069  050650  032737  002000  001344          BIT     #IAE,RMER11     ;WAS "IAE" DETECTED??
11070  050656  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
11071  050660  022737  001467  051216          CMP     #823.,500$      ;SHOULD AOE BE SET??
11072  050666  101025                          BHI     340$            ;NO!!
11073  050670  005737  051220                  TST     505$            ;MAYBE
11074  050674  001012                          BNE     335$            ;YES
11075  050676  005737  051222                  TST     510$
11076  050702  001007                          BNE     335$            ;YES !!
11077  050704  032737  000010  051224          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
11078  050712  001413                          BEQ     340$            ;NO !!
11079  050714  005737  001332                  TST     RMWC1           ;WAS ALL DATA TRANSFERRED ??
11080  050720  001410                          BEQ     340$            ;YES !!
11081  050722  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
11082  050730  005037  051220                  CLR     505$            ;CLEAR EXPECTED TRACK
11083  050734  012737  000001  051222          MOV     #1,510$ ;EXPECT SECTOR=1
11084  050742  013737  001344  001142  340$:   MOV     RMER11,$BDDAT   ;BAD DATA FOR TYPEOUT
11085  050750  042737  176777  001142          BIC     #^CAOE,$BDDAT
11086  050756  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
11087  050764  001413                          BEQ     345$            ;YES!!
11088  050766  062716  000004                  ADD     #4,(SP)
11089  050772  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
11090  051000  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11091  051004  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
11092  051006  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11093  051012  000240                          NOP
11094                                  ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
11095  051014  032737  002000  001344  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
11096  051022  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
11097  051024  013737  051220  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
11098  051032  000337  001140                  SWAB    $GDDAT
11099  051036  113737  051222  001140          MOVB    510$,$GDDAT
11100  051044  013737  001336  001142          MOV     RMDA1,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
11101  051052  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11102  051060  001413                          BEQ     350$            ;BRANCH IF EQUAL
11103  051062  062716  000004                  ADD     #4,(SP)
11104  051066  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
11105  051074  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11106  051100  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
11107  051102  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11108  051106  000240                          NOP
11109                          ;REPORT ERROR IF RMDC IS INCORRECT
11110  051110  013737  051216  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
11111  051116  042737  176000  001140          BIC     #^C1777,$GDDAT
11112  051124  013737  001364  001142          MOV     RMDC1,$BDDAT    ;SETUP RECEIVED CYLINDER
11113  051132  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
11114  051140  001413                          BEQ     355$            ;BRANCH IF EQUAL
11115  051142  062716  000004                  ADD     #4,(SP)
11116  051146  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
11117  051154  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11118  051160  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
11119  051162  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11120  051166  000240                          NOP
```

```
11121  051170  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11122  051174  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11123  051200  001403                          BEQ     360$
11124  051202  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
11125  051206  000402                          BR      365$
11126  051210  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11127  051214  000207                  365$:   RTS     PC
11128
11129  051216  000000                  500$:   .WORD   0               ;CYLINDER
11130  051220  000000                  505$:   .WORD   0               ;TRACK
11131  051222  000000                  510$:   .WORD   0               ;SECTOR
11132  051224  000000                  515$:   .WORD   0               ;FUNCTION CODE
11133
11134
```

```
11135                                   .SBTTL  DEVICE SELECT SUBROUTINE
11136
11137                                   ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
11138                                   ; TEST QUEUE.
11139
11140                                   ;CALL:
11141                                   ;(1)    JSR     PC,DEVSEL
11142                                   ;(2)    BR      ??              RETURN IF NO ERROR
11143                                   ;(3)    NOP                     RETURN IF ERROR
11144                                   ;(4)    ERROR                   ERROR DEFINED BY SUBROUTINE
11145
11146  051226                           DEVSEL:
11147
11148                                   ;CLEAR USER'S ERROR CALL
11149  051226  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11150  051232  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
11151  051236  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
11152                                   ;SAVE USER'S INFORMATION AND SETUP REGISTERS
11153  051242  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
11154  051246  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
11155  051252  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11156  051254  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11157  051256  012737  051376  000004          MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
11158  051264  012737  000300  000006          MOV     #PR6,ERRVEC+2
11159  051272  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
11160  051276  013701  001450                  MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER
11161
11162                                   ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
11163  051302  111160  000010                  MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
11164  051306  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
11165  051314  016037  000010  001174          MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
11166  051322  032737  010000  001174          BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
11167  051330  001407                          BEQ     10$             ;NO!!
11168  051332  062766  000004  000010          ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
11169  051340  112776  000111  000010          MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
11170  051346  000422                          BR      30$             ;
11171  051350  032737  004000  001176  10$:    BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
11172  051356  001021                          BNE     35$             ;YES!!
11173  051360  062766  000004  000010          ADD     #4,10(SP)
11174  051366  112776  000112  000010          MOVB    #112,@10(SP)
11175  051374  000407                          BR      30$
11176
11177  051376                           20$:
11178
11179                                   ;HANDLE BUS TIMEOUT
11180  051376  022626                          CMP     (SP)+,(SP)+     ;ADJUST SP
11181  051400  062766  000004  000010          ADD     #4,10(SP)
11182  051406  112776  000113  000010          MOVB    #113,@10(SP)
11183  051414  162766  000002  000010  30$:    SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR
11184
11185  051422                           35$:
11186
11187
11188                                   ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
11189  051422  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11190  051424  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
11191  051426  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
11192  051432  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
11193  051436  000207                          RTS     PC              ;EXIT
```

```
11194                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
11195                                   ;
11196                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
11197                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
11198
11199                                   ;
11200                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
11201                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
11202                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
11203                                   ;
11204                                   ;CALL:
11205                                   ;(1)    JSR     PC,SEKSTS
11206                                   ;       BR      ???             RETURN HERE IF NO ERROR
11207                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11208                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11209                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11210                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11211
11212  051440                           SEKSTS:
11213
11214                                   ;CLEAR USER'S ERROR  CALL
11215  051440  000240                          NOP
11216  051442  062716  000004                 ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11217  051446  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
11218  051452  162716  000004                 SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11219  051456  005037  052676                 CLR     300$            ;CLEAR ERROR FLAGS
11220
11221                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
11222                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
11223  051462  032737  000010  001344           BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
11224  051470  001424                           BEQ     1$              ;NO!!
11225  051472  032737  000010  001372           BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
11226  051500  001020                           BNE     1$              ;NOT SURE!!
11227
11228                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
11229  051502  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
11230  051506  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11231  051514  062716  000004                   ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
11232  051520  112776  000050  000000           MOVB    #50,@(SP)       ;ERROR #50
11233  051526  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11234  051532  004736                           JSR     PC,@(SP)+
11235  051534  162716  000010                   SUB     #10,(SP)        ;RESTORE STACK
11236  051540  000437                           BR      3$              ;IAE SHOULD  BE ZERO
11237
11238
11239                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
11240                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
11241  051542  012737  002000  001140   1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
11242  051550  052737  040000  052676           BIS     #SKI,300$       ;SET SKI ERROR FLAG
11243  051556  023727  001434  001466           CMP     RMDC0,#822.     ;CYLINDER > 822??
11244  051564  101025                           BHI     3$              ;YES!!
11245  051566  042737  040000  052676           BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
11246  051574  123727  001406  000037           CMPB    RMDA0,#31.      ;SECTOR > 31??
11247  051602  101016                           BHI     3$              ;YES!!
11248  051604  123727  001406  000035           CMPB    RMDA0,#29.      ;SECTOR > 29 ??
11249  051612  101404                           BLOS    2$              ;NO!!
```

```
11250  051614  032737  010000  001432          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
11251  051622  001406                          BEQ     3$              ;YES!!
11252  051624  123727  001407  000004  2$:     CMPB    RMDAO+1,#4.     ;TRACK  >4??
11253  051632  101002                          BHI     3$              ;YES!!
11254  051634  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
11255
11256                                  ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
11257  051640  013737  001344  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
11258  051646  042737  175777  001142          BIC     #^CIAE,$BDDAT
11259  051654  023737  001140  001142          CMP     $GDDAT,$BDDAT
11260  051662  001004                          BNE     35$             ;IAE IN ERROR
11261  051664  042737  040000  052676          BIC     #SKI,300$       ;CLEAR SKI FLAG
11262  051672  000413                          BR      5$              ;GO CHECK NEXT ERROR
11263  051674                          35$:
11264                                  ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
11265  051674  062716  000004                  ADD     #4,(SP)
11266  051700  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
11267  051706  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11268  051712  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
11269  051714  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11270  051720  000240                          NOP
11271  051722                          5$:
11272
11273                                  ;REPORT ANY IVC ERROR AS
11274                                  ;       IVC ERROR WITH VOLUME VALID ZERO
11275                                  ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
11276  051722  032737  010000  001372          BIT     #IVC,RMER2I     ;IVC ERROR??
11277  051730  001427                          BEQ     52$             ;NO!!
11278  051732  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11279  051736  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11280  051744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11281  051750  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
11282  051756  032737  000100  001342          BIT     #VV,RMDSI
11283  051764  001403                          BEQ     51$
11284  051766  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
11285  051774  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11286  052000  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11287  052002  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
11288  052006  000240                          NOP
11289
11290  052010  013737  001372  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11291  052016  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
11292  052024  013737  052676  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
11293  052032  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
11294  052040  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
11295
11296                                  ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
11297  052042  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
11298  052050  001032                          BNE     54$             ;YES !!
11299  052052  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11300  052056  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
11301  052064  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
11302  052070  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11303  052072  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11304  052076  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
11305  052100                          53$:
```

```
11306
11307                                   ;REPORT ERROR IF SKI IS SET
11308  052100  032737  040000  001142           BIT     #SKI,$BDDAT     ;IS SKI SET ??
11309  052106  001413                           BEQ     54$             ;NO - SKI IS OK
11310  052110  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11311  052114  112776  000054  000000           MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
11312  052122  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11313  052126  004736                           JSR     PC,@(SP)+       ;REPORT SEEK ERROR
11314  052130  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
11315  052134  000240                           NOP
11316
11317                                   ;REPORT ANY DEVICE CHECK
11318  052136  032737  000200  001372  54$:     BIT     #DVC,RMER21     ;WAS THERE DVC DURING SEEK??
11319  052144  001420                           BEQ     6$              ;NO!!
11320  052146  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
11321  052152  013737  001372  001142           MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11322  052160  062716  000004                   ADD     #4,(SP)
11323  052164  112776  000055  000000           MOVB    #55,@(SP)       ;ERROR  #55
11324  052172  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11325  052176  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11326  052200  162716  000010                   SUB     #10,(SP)        ;RESTORE SP
11327  052204  000240                           NOP
11328
11329                                   ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
11330                                   ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
11331  052206  032737  020000  001344  6$:      BIT     #OPI,RMER11     ;"OPI" ERROR??
11332  052214  001427                           BEQ     8$              ;NO!!
11333  052216  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
11334  052222  013737  001344  001142           MOV     RMER11,$BDDAT   ;RECEIVED STATUS
11335  052230  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11336  052234  112776  000052  000000           MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
11337  052242  032737  010000  001342           BIT     #MOL,RMDS1      ;WAS MEDIUM ON LINE??
11338  052250  001403                           BEQ     7$              ;NO!!
11339  052252  112776  000053  000000           MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
11340  052260  162716  000002           7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11341  052264  004736                           JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
11342  052266  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
11343  052272  000240                           NOP
11344
11345                                   ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
11346  052274  013746  001342           8$:     MOV     RMDS1,-(SP)
11347  052300  042716  047677                   BIC     #^C<ATA!PIP!MOL!VV>,(SP)
11348  052304  022726  110100                   CMP     #ATA!MOL!VV,(SP)+
11349  052310  001002                           BNE     9$              ;ERROR IN RMDS
11350  052312  000137  052646                   JMP     14$             ;RMDS IS OK
11351
11352                                   ;REPORT ERROR IF MOL = 0 AND OPI = 0
11353  052316  032737  010000  001342  9$:      BIT     #MOL,RMDS1      ;IS MOL RESET??
11354  052324  001030                           BNE     10$             ;NO - MOL IS SET
11355  052326  032737  020000  001344           BIT     #OPI,RMER11     ;WAS OPI SET
11356  052334  001024                           BNE     10$             ;YES - DONT REPORT ERROR
11357  052336  013737  001342  001140           MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11358  052344  052737  010000  001140           BIS     #MOL,$GDDAT
11359  052352  013737  001342  001142           MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11360  052360  062716  000004                   ADD     #4,(SP)
11361  052364  112776  000062  000000           MOVB    #62,@(SP)
```

```
11362  052372  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11363  052376  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
11364  052400  162716  000010            SUB     #10,(SP)
11365  052404  000240                    NOP
11366
11367                                ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
11368  052406  032737  020000  001342  10$:    BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
11369  052414  001430                    BEQ     11$              ;NO!!
11370  052416  032737  040000  001372        BIT     #SKI,RMER2I      ;WAS "SKI"SET??
11371  052424  001024                    BNE     11$              ;YES-DONT REPORT PIP
11372  052426  013737  001342  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11373  052434  042737  020000  001142        BIC     #PIP,$BDDAT
11374  052442  013737  001342  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11375  052450  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11376  052454  112776  000056  000000        MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
11377  052462  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11378  052466  004736                    JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
11379  052470  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
11380  052474  000240                    NOP
11381
11382                                ;REPORT AN ERROR IF "ATA" IS NOT SET
11383  052476  032737  100000  001342  11$:    BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
11384  052504  001024                    BNE     13$              ;YES!!
11385  052506  013737  001342  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11386  052514  052737  110600  001140        BIS     #ATA!MOL!DPR!DRY,$GDDAT
11387  052522  013737  001342  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11388  052530  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11389  052534  112776  000057  000000        MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
11390  052542  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11391  052546  004736                    JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
11392                                                             ;SEEK TEST
11393  052550  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
11394  052554  000240                    NOP
11395
11396                                ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
11397  052556  032737  000100  001342  13$:    BIT     #VV,RMDSI        ;IS VV = 0 ??
11398  052564  001030                    BNE     14$              ;NO!!
11399  052566  032737  010000  001372        BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
11400  052574  001024                    BNE     14$              ;NO - IVC IS SET
11401  052576  013737  001342  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11402  052604  052737  000100  001140        BIS     #VV,$GDDAT
11403  052612  013737  001342  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11404  052620  062716  000004            ADD     #4,(SP)
11405  052624  112776  000064  000000        MOVB    #64,@(SP)        ;ERROR #64
11406  052632  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11407  052636  004736                    JSR     PC,@(SP)+
11408  052640  162716  000010            SUB     #10,(SP)
11409  052644  000240                    NOP
11410  052646                        14$:
11411
11412                                ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
11413  052646  000240                    NOP
11414  052650  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
11415  052654  105776  000000            TSTB    @(SP)            ;WAS ERROR CALLED??
11416  052660  001403                    BEQ     15$              ;NO!!
11417  052662  062716  000004            ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
11418  052666  000402                    BR      16$
11419
11420  052670  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11421  052674  000207              16$:    RTS     PC              ;RETURN
11422
11423  052676  000000              300$:   .WORD   0               ;ERROR FLAGS
11424
```

```
11425                                   .SBTTL  CONTROLLER CLEAR SUBROUTINE
11426
11427                                   ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
11428                                   ;AND DRIVES, THEN SELECTS THE DRIVE.
11429                                   ;
11430                                   ;CALL:  JSR     PC,CNTCLR
11431                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
11432                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
11433                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
11434                                   ;       ???
11435
11436  052700                           CNTCLR:
11437  052700  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11438  052702  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11439  052704  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
11440  052710  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
11441  052714  012737  052754  000004          MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
11442  052722  012737  000300  000006          MOV     #PR6,ERRVEC+2
11443  052730  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
11444  052734  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
11445  052742  013701  001450                  MOV     TSTQUE,R1
11446  052746  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
11447  052752  000412                          BR      20$
11448  052754  022626                  10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
11449  052756  062766  000004  000010          ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
11450  052764  112776  000007  000010          MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
11451  052772  162766  000002  000010          SUB     #2,10(SP)
11452  053000                           20$:
11453  053000  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
11454  053004  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
11455  053010  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11456  053012  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11457  053014  000207                          RTS     PC
11458
```

```
11459                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
11460
11461                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
11462                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
11463                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
11464                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
11465
11466                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
11467                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
11468                                   ;
11469                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
11470                                   ;
11471
11472                                   ;CALL:
11473                                   ;(1)    JSR     PC,CLRSTS
11474                                   ;       BR      ???             RETURN HERE IF NO ERROR
11475                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11476                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11477                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11478                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11479
11480  053016                           CLRSTS:
11481
11482                                   ;CLEAR USER'S ERROR CALL
11483  053016  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR
11484  053022  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
11485  053026  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
11486                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
11487  053032  013737  001330  001142  4$:     MOV     RMCS11,$BDDAT   ;VERIFY RMCS1
11488  053040  042737  100000  001142          BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
11489  053046  012737  004200  001140          MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
11490  053054  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
11491  053062  001413                          BEQ     5$              ;BRANCH IF EQUAL
11492  053064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11493  053070  112776  000126  000000          MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
11494  053076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11495  053102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11496  053104  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11497  053110  000240                          NOP
11498                                   ;REPORT ERROR IF RMBA NOT RESET
11499  053112  005037  001140          5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
11500  053116  013737  001334  001142          MOV     RMBA1,$BDDAT
11501  053124  001413                          BEQ     7$              ;BRANCH IF ZERO
11502  053126  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11503  053132  112776  000127  000000          MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
11504  053140  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11505  053144  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11506  053146  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11507  053152  000240                          NOP
11508                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
11509  053154  013737  001340  001142  7$:     MOV     RMCS21,$BDDAT   ;VERIFY RMCS2
11510  053162  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11511  053164  005046                          CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
11512  053166  013701  001450                  MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
11513  053172  111116                          MOVB    (R1),(SP)
11514  053174  052716  000100                  BIS     #IR,(SP)
```

```
11515  053200  012637  001140          MOV     (SP)+,$GDDAT
11516  053204  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
11517  053206  023737  001140  001142  CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11518  053214  001413                  BEQ     9$              ;BRANCH IF EQUAL
11519  053216  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11520  053222  112776  000130  000000  MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
11521  053230  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11522  053234  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11523  053236  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11524  053242  000240                  NOP
11525                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
11526  053244  005037  001140  9$:     CLR     $GDDAT          ;VERIFY RMER1
11527  053250  013737  001344  001142  MOV     RMER11,$BDDAT
11528  053256  042737  040000  001142  BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
11529  053264  001413                  BEQ     13$             ;BRANCH IF ZERO
11530  053266  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11531  053272  112776  000131  000000  MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
11532  053300  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11533  053304  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11534  053306  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11535  053312  000240                  NOP
11536                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
11537  053314  013737  001354  001142  13$:     MOV     RMMR11,$BDDAT   ;VERIFY RMMR
11538  053322  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
11539  053330  012737  000010  001140          MOV     #WD,$GDDAT      ;EXPECT WRITE DATA BIT
11540  053336  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11541  053344  001413                          BEQ     17$             ;BRANCH IF 0
11542  053346  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11543  053352  112776  000133  000000          MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
11544  053360  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11545  053364  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11546  053366  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11547  053372  000240                          NOP
11548                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
11549  053374  005037  001140  17$:     CLR     $GDDAT          ;EXPECT ZEROS
11550  053400  013737  001376  001142  MOV     RMEC21,$BDDAT   ;VERIFY RMEC2=0
11551  053406  001413                  BEQ     19$
11552  053410  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11553  053414  112776  000135  000000  MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
11554  053422  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11555  053426  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11556  053430  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11557  053434  000240                  NOP
11558                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
11559  053436  013737  001370  001142  19$:     MOV     RMMR21,$BDDAT   ;VERIFY RMMR2
11560  053444  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
11561  053452  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
11562  053460  023737  001140  001142          CMP     $GDDAT,$BDDAT
11563  053466  001413                          BEQ     21$
11564  053470  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11565  053474  112776  000136  000000          MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
11566  053502  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11567  053506  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11568  053510  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11569  053514  000240                          NOP
11570                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
11571  053516  005037  001140          21$:    CLR     $GDDAT          ;EXPECT ALL ZEROS
11572  053522  013737  001372  001142          MOV     RMER21,$BDDAT   ;VERIFY RMER2
11573  053530  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
11574  053536  001413                          BEQ     215$            ;BRANCH IF OTHER BITS 0
11575  053540  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11576  053544  112776  000174  000000          MOVB    #174,@(SP)      ;WITE ERROR NUMBER IN CALL
11577  053552  162716  000002                  SUB     #2.(SP)         ;MOVE SP TO RETURN FOR ERROR
11578  053556  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11579  053560  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11580  053564  000240                          NOP
11581                                  ;REPORT ERROR IF RMDS NOT INITIALIZED
11582  053566  013737  001342  001142  215$:   MOV     RMDS1,$BDDAT    ;TEST DRIVE STATUS REGISTER
11583  053574  042737  177177  001142          BIC     #^C<DRY!DPR>,$BDDAT
11584  053602  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
11585  053610  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11586  053616  001413                          BEQ     22$             ;BRANCH IF EQUAL
11587  053620  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11588  053624  112776  000134  000000          MOVB    #134,@(SP)      ;WRITE ERROR NUMBER
11589  053632  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11590  053636  004736                          JSR     PC,@(SP)+       ;REPORT ERROR TO USER
11591  053640  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11592  053644  000240                          NOP
11593  053646  062716  000004          22$:    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11594  053652  105776  000000                  TSTB    @(SP)           ;WAS AN ERROE DETECTED??
11595  053656  001403                          BEQ     23$             ;NO!!
11596  053660  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11597  053664  000402                          BR      24$
11598  053666  162716  000004          23$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11599  053672  000240                  24$:    NOP
11600  053674  000207                          RTS     PC
```

```
11601                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
11602
11603                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
11604                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
11605                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
11606
11607                                   ;CALL:
11608                                   ;(1)    JSR     PC,ACKSTS
11609                                   ;       BR      ???             RETURN HERE IF NO ERROR
11610                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11611                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11612                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11613                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11614
11615  053676                           ACKSTS:
11616
11617                                   ;CLEAR USER'S ERROR CALL
11618  053676  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11619  053702  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
11620  053706  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
11621
11622                                   ;REPORT AN ERROR  IF "VV" IS 0
11623  053712  032737  000100  001342          BIT     #VV,RMDS1       ;IS VOLUME VALID SET??
11624  053720  001024                          BNE     1$              ;YES!!
11625  053722  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11626  053730  052737  000100  001140          BIS     #VV,$GDDAT
11627  053736  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11628  053744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11629  053750  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
11630  053756  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11631  053762  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
11632  053764  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11633  053770  000240                          NOP
11634  053772                           1$:
11635
11636                                   ;REPORT AN ERROR IF "MOL" IS 0
11637  053772  032737  010000  001342          BIT     #MOL,RMDS1      ;IS MOL SET??
11638  054000  001024                          BNE     2$              ;YES!!
11639  054002  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11640  054010  052737  010000  001140          BIS     #MOL,$GDDAT
11641  054016  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11642  054024  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11643  054030  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11644  054036  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11645  054042  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
11646  054044  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
11647  054050  000240                          NOP
11648  054052                           2$:
11649
11650                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
11651  054052  032737  060007  001344          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
11652  054060  001570                          BEQ     7$
11653
11654                                   ;REPORT AN ERROR IF "UNS" IS SET
11655  054062  032737  040000  001344          BIT     #UNS,RMER1I     ;WAS UNS SET??
11656  054070  001424                          BEQ     3$              ;NO!!
```

```
 11657  054072  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
 11658  054100  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
 11659  054106  042737  040000  001140          BIC     #UNS,$GDDAT
 11660  054114  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 11661  054120  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 11662  054126  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11663  054132  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 11664  054134  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 11665  054140  000240                          NOP
 11666  054142                          3$:
 11667
 11668                                  ;REPORT ANY OPI ERROR
 11669  054142  032737  020000  001344          BIT     #OPI,RMER11     ;WAS OPI SET ??
 11670  054150  001424                          BEQ     4$              ;NO!!
 11671  054152  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
 11672  054160  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
 11673  054166  042737  020000  001140          BIC     #OPI,$GDDAT
 11674  054174  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 11675  054200  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 11676  054206  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11677  054212  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 11678  054214  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 11679  054220  000240                          NOP
 11680  054222                          4$:
 11681
 11682                                  ;REPORT ANY RMR ERROR
 11683  054222  032737  000004  001344          BIT     #RMR,RMER11     ;WAS RMR SET??
 11684  054230  001424                          BEQ     5$              ;NO!!
 11685  054232  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
 11686  054240  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
 11687  054246  042737  000004  001140          BIC     #RMR,$GDDAT
 11688  054254  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 11689  054260  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 11690  054266  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11691  054272  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 11692  054274  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 11693  054300  000240                          NOP
 11694  054302                          5$:
 11695
 11696                                  ;REPORT ANY     ILR ERROR
 11697  054302  032737  000002  001344          BIT     #ILR,RMER11     ;WAS ILR SET??
 11698  054310  001424                          BEQ     6$              ;NO!!
 11699  054312  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
 11700  054320  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
 11701  054326  042737  000002  001140          BIC     #ILR,$GDDAT
 11702  054334  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 11703  054340  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 11704  054346  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11705  054352  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 11706  054354  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 11707  054360  000240                          NOP
 11708  054362                          6$:
 11709
 11710                                  ;REPORT ANY     ILF ERROR
 11711  054362  032737  000001  001344          BIT     #ILF,RMER11     ;WAS ILF SET??
 11712  054370  001424                          BEQ     7$              ;NO!!
```

```
11713  054372  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
11714  054400  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
11715  054406  042737  000001  001140          BIC     #1LF,$GDDAT
11716  054414  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11717  054420  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11718  054426  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11719  054432  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11720  054434  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11721  054440  000240                          NOP
11722  054442                          7$:
11723
11724                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
11725  054442  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11726  054446  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11727  054452  001403                          BEQ     8$              ;NO!!
11728  054454  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11729  054460  000402                          BR      9$
11730  054462  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11731  054466  000240                  9$:     NOP
11732  054470  000207                          RTS     PC
11733
```

```
11734                                   .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
11735
11736                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
11737                                   ;USING THE STATUS STORED IN THE GET BUFFER.
11738
11739                                   ;CALL:
11740                                   ;
11741                                   ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
11742                                   ;       BR      ???             RETURN HERE IF NO ERROR
11743                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11744                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11745                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11746                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11747
11748  054472                           RCLSTS:
11749
11750                                   ;CLEAR USER'S ERROR NUMBER
11751  054472  062716  000004                   ADD     #4,(SP)
11752  054476  105076  000000                   CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
11753  054502  162716  000004                   SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
11754
11755
11756                                   ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
11757  054506  032737  022011  001344           BIT     #OPI!PAR!ILF!IAE,RMER1I
11758  054514  001553                           BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
11759
11760                                   ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
11761                                   ;"PAR" = 1 AND "DPE" = 0
11762  054516  032737  000010  001344           BIT     #PAR,RMER1I     ;WAS "PAR" SET??
11763  054524  001430                           BEQ     1$              ;NO!!
11764  054526  032737  000010  001372           BIT     #DPE,RMER2I     ;WAS "DPE" SET??
11765  054534  001024                           BNE     1$              ;YES - NOT A REGISTER ERROR
11766  054536  013737  001344  001140           MOV RMER1I,$GDDAT       ;EXPECTED STATUS
11767  054544  042737  000010  001140           BIC     #PAR,$GDDAT
11768  054552  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11769  054560  062716  000004                   ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11770  054564  112776  000050  000000           MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
11771  054572  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11772  054576  004736                           JSR     PC,@(SP)+       ;GO REPORT ERROR
11773  054600  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11774  054604  000240                           NOP
11775  054606                           1$:
11776
11777                                   ;REPORT ANY "ILF" ERROR
11778  054606  032737  000001  001344           BIT     #ILF,RMER1I     ;WAS "ILF" SET??
11779  054614  001424                           BEQ     2$              ;NO!!
11780  054616  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11781  054624  042737  000001  001140           BIC     #ILF,$GDDAT
11782  054632  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11783  054640  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11784  054644  112776  000071  000000           MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
11785  054652  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11786  054656  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11787  054660  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11788  054664  000240                           NOP
11789  054666                           2$:
```

```
11790
11791                                   ;REPORT ANY "OPI" ERROR AS
11792                                   ;       . OPI DUE TO "MOL" = 0
11793                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
11794  054666  032737  020000  001344           BIT     #OPI,RMER1I     ;WAS OPI SET??
11795  054674  001433                           BEQ     31$             ;NO!!
11796  054676  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11797  054704  042737  020000  001140           BIC     #OPI,$GDDAT
11798  054712  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11799  054720  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11800  054724  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11801  054732  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
11802  054740  001403                           BEQ     3$              ;YES!!
11803  054742  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
11804  054750  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11805  054754  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11806  054756  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11807  054762  000240                           NOP
11808  054764                           31$:
11809
11810                                   ;REPORT AN ERROR IF "IAE" IS SET
11811  054764  032737  002000  001344           BIT     #IAE,RMER1I     ;IS "IAE" SET??
11812  054772  001424                           BEQ     4$              ;NO!!
11813  054774  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11814  055002  042737  002000  001140           BIC     #IAE,$GDDAT
11815  055010  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11816  055016  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11817  055022  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11818  055030  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11819  055034  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
11820  055036  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
11821  055042  000240                           NOP
11822  055044                           4$:
11823
11824                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
11825  055044  032737  050200  001372           BIT     #SKI!IVC!DVC,RMER2I
11826  055052  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
11827
11828
11829                                   ;REPORT ANY "IVC" ERROR AS
11830                                   ;       .IVC WITH VV = 0
11831                                   ;       .ERRONEOUS IVC ERROR
11832  055054  032737  010000  001372           BIT     #IVC,RMER2I     ;WAS IVC SET??
11833  055062  001433                           BEQ     6$              ;NO!!
11834  055064  013737  001372  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11835  055072  042737  010000  001140           BIC     #IVC,$GDDAT
11836  055100  013737  001372  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11837  055106  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11838  055112  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
11839  055120  032737  000100  001342           BIT     #VV,RMDSI       ;WAS VV = 0??
11840  055126  001403                           BEQ     5$              ;YES!!
11841  055130  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
11842  055136  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11843  055142  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11844  055144  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11845  055150  000240                           NOP
```

```
11846  055152                                  6$:
11847
11848                                          ;REPORT ANY "SKI" ERROR
11849  055152  032737  040000  001372          BIT     #SKI,RMER21     ;WAS SKI SET??
11850  055160  001424                          BEQ     7$              ;NO!!
11851  055162  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
11852  055170  042737  040000  001140          BIC     #SKI,$GDDAT
11853  055176  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11854  055204  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11855  055210  112776  000076  000000          MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
11856  055216  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11857  055222  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11858  055224  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
11859  055230  000240                          NOP
11860  055232                                  7$:
11861
11862                                          ;REPORT ANY "DVC" ERROR
11863  055232  032737  000200  001372          BIT     #DVC,RMER21     ;WAS "DVC" SET??
11864  055240  001424                          BEQ     8$              ;NO!!
11865  055242  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
11866  055250  042737  000200  001140          BIC     #DVC,$GDDAT
11867  055256  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11868  055264  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11869  055270  112776  000077  000000          MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
11870  055276  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11871  055302  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11872  055304  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11873  055310  000240                          NOP
11874  055312                                  8$:
11875
11876                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
11877  055312  013746  001342                  MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
11878  055316  042716  047676                  BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
11879  055322  022726  110100                  CMP     #ATA!MOL!VV,(SP)+
11880  055326  001002                          BNE     85$
11881  055330  000137  055744                  JMP     13$
11882  055334                                  85$:
11883
11884                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
11885                                          ;LINE AFTER RECALIBRATE WAS INITIATED
11886  055334  032737  010000  001342          BIT     #MOL,RMDSI      ;DID MOL DROP??
11887  055342  001030                          BNE     9$              ;NO!!
11888  055344  032737  020000  001344          BIT     #OPI,RMER11     ;WAS OPI ERROR REPORTED??
11889  055352  001024                          BNE     9$              ;YES - DON'T REPORT MOL=0
11890  055354  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11891  055362  052737  010000  001140          BIS     #MOL,$GDDAT
11892  055370  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11893  055376  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11894  055402  112776  000100  000000          MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
11895  055410  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11896  055414  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11897  055416  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11898  055422  000240                          NOP
11899  055424                                  9$:
11900
11901                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
11902  055424  032737  000100  001342          BIT     #VV,RMDSI       ;DID "VV" DROP??
11903  055432  001030                          BNE     10$             ;NO!!
11904  055434  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
11905  055442  001024                          BNE     10$             ;YES - DONT REPORT VV=0
11906  055444  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11907  055452  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11908  055460  052737  000100  001140          BIS     #VV,$GDDAT
11909  055466  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11910  055472  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
11911  055500  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11912  055504  004736                          JSR     PC,@(SP)+
11913  055506  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USEP'S BRANCH
11914  055512  000240                          NOP
11915  055514                          10$:
11916
11917                                  ,REPORT AN ERROR IF ATA IS NOT SET
11918  055514  032737  100000  001342          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
11919  055522  001024                          BNE     11$             ;YES!!
11920  055524  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11921  055532  052737  100000  001140          BIS     #ATA,$GDDAT
11922  055540  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11923  055546  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11924  055552  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
11925  055560  162716  000002                  SUB     #2,(SP)
11926  055564  004736                          JSR     PC,@(SP)+
11927  055566  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11928  055572  000240                          NOP
11929
11930  055574                          11$:
11931
11932                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
11933                                  ;ALWAYS CLEAR OFFSET MODE
11934  055574  032737  000001  001342          BIT     #OM,RMDSI       ;WAS "OM" RESET??
11935  055602  001424                          BEQ     12$             ;YES!!
11936  055604  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11937  055612  042737  000001  001140          BIC     #OM,$GDDAT
11938  055620  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11939  055626  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11940  055632  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
11941  055640  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11942  055644  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11943  055646  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11944  055652  000240                          NOP
11945  055654                          12$:
11946
11947                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
11948                                  ;CYLINDER
11949  055654  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11950  055662  001430                          BEQ     13$             ;NO!!
11951  055664  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11952  055672  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
11953  055674  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11954  055702  042737  020000  001140          BIC     #PIP,$GDDAT
11955  055710  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11956  055716  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11957  055722  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
11958  055730  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11959  055734  004736                    JSR     PC,@(SP)+
11960  055736  162716  000010            SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
11961  055742  000240                    NOP
11962  055744                        13$:
11963
11964                                    ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
11965  055744  032737  040006  001344    BIT     #ILR!RMR!UNS,RMER1I
11966  055752  001514                    BEQ     16$
11967
11968                                    ;REPORT AN ERROR IF "ILR" IS SET
11969  055754  032737  000002  001344    BIT     #ILR,RMER1I      ;WAS ILR SET DURING RECALIBRATE??
11970  055762  001424                    BEQ     14$              ;NO!!
11971  055764  013737  001344  001140    MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11972  055772  042737  000002  001140    BIC     #ILR,$GDDAT
11973  056000  013737  001344  001142    MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11974  056006  062716  000004            ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11975  056012  112776  000105  000000    MOVB    #105,@(SP)       ;WRITE ERROR NUMBER IN CALL
11976  056020  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11977  056024  004736                    JSR     PC,@(SP)+
11978  056026  162716  000010            SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
11979  056032  000240                    NOP
11980  056034                        14$:
11981
11982                                    ;REPORT AN ERROR IF "RMR" IS SET
11983  056034  032737  000004  001344    BIT     #RMR,RMER1I      ;WAS RMR SET??
11984  056042  001424                    BEQ     15$              ;NO!!
11985  056044  013737  001344  001140    MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11986  056052  042737  000004  001140    BIC     #RMR,$GDDAT
11987  056060  013737  001344  001142    MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11988  056066  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11989  056072  112776  000106  000000    MOVB    #106,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11990  056100  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11991  056104  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
11992  056106  162716  000010            SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
11993  056112  000240                    NOP
11994  056114                        15$:
11995
11996                                    ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
11997  056114  032737  040000  001344    BIT     #UNS,RMER1I      ;WAS UNSAFE ON??
11998  056122  001430                    BEQ     16$              ;NO!!
11999  056124  032737  000200  001372    BIT     #DVC,RMER2I      ;WAS THERE A DEVICE CHECK??
12000  056132  001024                    BNE     16$              ;YES - DON'T REPORT UNSAFE
12001  056134  013737  001344  001140    MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
12002  056142  042737  040000  001140    BIC     #UNS,$GDDAT
12003  056150  013737  001344  001142    MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
12004  056156  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
12005  056162  112776  000107  000000    MOVB    #107,@(SP)       ;WRITE ERROR NUMBER
12006  056170  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
12007  056174  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
12008  056176  162716  000010            SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
12009  056202  000240                    NOP
12010  056204                        16$:
12011
12012                                    ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12013  056204  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
```

```
12014  056210  105776  000000          TSTB    @(SP)             ;WAS AN ERROR REPORTED??
12015  056214  001403                  BEQ     17$               ;NO!!
12016  056216  062716  000004          ADD     #4,(SP)           ;YES - AUGMENT SP RETURN
12017  056222  000402                  BR      18$
12018  056224  162716  000004  17$:    SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
12019  056230  000240          18$:    NOP
12020  056232  000207                  RTS     PC                ;STATUS CECK IS COMPLETE
12021
```

```
12022                                   .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
12023                                   ;       BR      ???             RETURN HERE IF NO ERROR
12024                                   ;       NOP                     RETURN HERE TO REPORT AN ERRO
12025                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12026                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12027                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12028
12029  056234                           DRVSTS:
12030
12031                                   ;CLEAR USER'S ERROR CALL
12032  056234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12033  056240  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
12034  056244  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
12035                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
12036  056250  013737  001330  001142  4$:     MOV     RMCS11,$BDDAT   ;CHECK RMCS1
12037  056256  042737  173700  001142          BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
12038  056264  012737  004010  001140          MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
12039  056272  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12040  056300  001443                          BEQ     6$              ;BRANCH IF EQUAL
12041  056302  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12042  056306  112776  000141  000000          MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12043  056314  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12044  056320  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12045  056322  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12046  056326  000240                          NOP
12047                                   ;REPORT ERROR IF RMDS NOT INITIALIZED
12048  056330  013737  001342  001142  5$:     MOV     RMDS1,$BDDAT    ;CHECK RMDS
12049  056336  042737  021101  001142          BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
12050  056344  012737  010600  001140          MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
12051  056352  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12052  056360  001413                          BEQ     6$              ;BRANCH IF EQUAL
12053  056362  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12054  056366  112776  000142  000000          MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12055  056374  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12056  056400  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12057  056402  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12058  056406  000240                          NOP
12059                                   ;REPORT ERROR IF RMER1 NOT INITIALIZED
12060  056410  005037  001140          6$:     CLR     $GDDAT          ;EXPECT 0'S
12061  056414  013737  001344  001142          MOV     RMER11,$BDDAT   ;CHECK RMER1
12062  056422  001413                          BEQ     8$              ;BRANCH IF EQUAL
12063  056424  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12064  056430  112776  000143  000000          MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12065  056436  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12066  056442  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12067  056444  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12068  056450  000240                          NOP
12069                                   ;REPORT ERROR IF ATA NOT INITIALIZED
12070  056452  013737  001346  001142  8$:     MOV     RMAS1,$BDDAT    ;CHECK ATTENTION BIT
12071  056460  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12072  056462  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12073  056464  013701  001450                  MOV     TSTQUE,R1
12074  056470  116102  000001                  MOVB    1(R1),R2
12075  056474  042702  177400                  BIC     #^CATNMSK,R2
12076  056500  005102                          COM     R2
12077  056502  040237  001142                  BIC     R2,$BDDAT
```

```
12078  056506  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
12079  056510  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
12080  056512  005737  001142                TST     $BDDAT          ;IS ATTENTION CLEARED??
12081  056516  001413                        BEQ     9$              ;BRANCH IF ATTENTION CLEARED
12082  056520  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12083  056524  112776  000144  000000        MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12084  056532  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12085  056536  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12086  056540  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12087  056544  000240                        NOP
12088                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
12089  056546  013737  001354  001142  9$:   MOV     RMMR1I,$BDDAT   ;CHECK RMMR
12090  056554  042737  000046  001142        BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
12091  056562  012737  000010  001140        MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
12092  056570  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
12093  056576  001413                        BEQ     11$             ;BRANCH IF ZERO
12094  056600  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12095  056604  112776  000145  000000        MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12096  056612  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12097  056616  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12098  056620  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12099  056624  000240                        NOP
12100                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
12101  056626  013737  001370  001142  11$:   MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
12102  056634  042737  140000  001142        BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
12103  056642  012737  011777  001140        MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
12104  056650  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12105  056656  001413                        BEQ     15$             ;BRANCH IF EQUAL
12106  056660  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12107  056664  112776  000146  000000        MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12108  056672  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12109  056676  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12110  056700  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12111  056704  000240                        NOP
12112  056706  005037  001140          15$:  CLR     $GDDAT          ;EXPECT ZEROS
12113                          ;REPORT ERROR IF RMEC2 NOT RESET
12114  056712  013737  001376  001142        MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
12115  056720  001413                        BEQ     17$             ;BRANCH IF 0
12116  056722  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12117  056726  112776  000150  000000        MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12118  056734  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12119  056740  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12120  056742  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12121  056746  000240                        NOP
12122                          ;REPORT ERROR IF RMER2 NOT RESET
12123  056750  013737  001372  001142  17$:   MOV     RMER2I,$BDDAT   ;CHECK RMER2
12124  056756  001413                        BEQ     18$             ;BRANCH IF NO ERROR
12125  056760  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12126  056764  112776  000147  000000        MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12127  056772  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12128  056776  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12129  057000  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12130  057004  000240                        NOP
12131  057006                          18$:
12132
12133  057006                          19$:
```

```
12134
12135                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
12136  057006  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12137  057012  105776  000000                TSTB    @(SP)           ;WAS AN ERROR DETECTED??
12138  057016  001403                         BEQ     21$             ;NO!!
12139  057020  062716  000004                ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12140  057024  000402                         BR      23$
12141  057026  162716  000004          21$:  SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
12142  057032  000240                   23$:  NOP
12143  057034  000207                         RTS     PC              ;RETURN TO USER
```

```
12144                                   .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
12145
12146                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
12147                                   ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
12148                                   ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
12149                                   ;THE ERROR NUMBER IN THE USERS ERROR CALL.
12150
12151                                   ;USER'S SUBROUTINE CALL:
12152                                   ;(1)    JSR     PC,DTASTS
12153                                   ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
12154                                   ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
12155                                   ;(4)    ERROR                   SUB WRITES ERROR NUMBER
12156                                   ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
12157                                   ;(6)    ??                      SUB RETURNS HERE AFTER ALL
12158                                   ;                               ERRORS ARE REPORTED
12159
12160  057036                           DTASTS:
12161
12162                                   ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
12163  057036  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12164  057042  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
12165  057046  162716  000004                  SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
12166  057052  005037  062432                  CLR     500$            ;CLEAR ERROR FLAGS
12167
12168                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
12169  057056  032737  020000  001330          BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
12170  057064  001422                          BEQ     10$             ;NO!!
12171  057066  013737  001330  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
12172  057074  042737  020000  001140          BIC     #MCPE,$GDDAT
12173  057102  013737  001330  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
12174  057110  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12175  057114  112776  000013  000000          MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
12176  057122  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12177  057126  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12178  057130  000466                          BR      30$
12179  057132                           10$:
12180
12181                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
12182                                   ;PAR=1  AND DPE = 0
12183  057132  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
12184  057140  001435                          BEQ     20$             ;NO!!
12185  057142  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
12186  057150  001031                          BNE     20$             ;NO!!
12187  057152  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12188  057160  042737  000010  001140          BIC     #PAR,$GDDAT
12189  057166  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12190  057174  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12191  057200  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
12192  057206  032737  001000  001330          BIT     #MXF,RMCS1I     ;DID MXF GET SET??
12193  057214  001003                          BNE     15$             ;YES!!
12194  057216  112776  000274  000000          MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
12195  057224  162716  000002           15$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12196  057230  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12197  057232  000425                          BR      30$
12198
12199  057234                           20$:
```

```
12200
12201                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
12202                                   ;MECHANICAL POSITIONING
12203
12204                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
12205                                   ;CODE AND GO BIT WERE LOADED
12206  057234  032737  001000  001340          BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
12207  057242  001425                        BEQ     40$             ;NO!!
12208  057244  013737  001340  001140        MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12209  057252  042737  001000  001140        BIC     #MXF,$GDDAT
12210  057260  013737  001340  001142        MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12211  057266  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12212  057272  112776  000275  000000        MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
12213  057300  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12214  057304  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12215  057306                          30$:
12216
12217                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
12218  057306  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12219  057312  000137  062404                JMP     380$            ;SKIP TO END OF SUB
12220
12221  057316                          40$:
12222
12223                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
12224                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
12225                                   ;"MOL"
12226  057316  032737  020000  001344          BIT     #OPI,RMER1I     ;IS "OPI" SET??
12227  057324  001447                        BEQ     60$             ;NO!!
12228  057326  013737  001344  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12229  057334  042737  020000  001140        BIC     #OPI,$GDDAT
12230  057342  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12231  057350  032737  010000  001342        BIT     #MOL,RMDS1      ;WAS MEDIUM OFF LINE??
12232  057356  001404                        BEQ     45$             ;YES!!
12233  057360  032737  000100  001342        BIT     #VV,RMDS1       ;WAS "MOL" INTERMITTENT??
12234  057366  001013                        BNE     50$             ;NO!!
12235  057370  062716  000004          45$:  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12236  057374  112776  000276  000000        MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
12237  057402  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12238  057406  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12239  057410  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12240  057414  000413                        BR      60$
12241  057416                          50$:
12242
12243                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
12244                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
12245  057416  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12246  057422  112776  000277  000000        MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
12247  057430  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12248  057434  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12249  057436  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12250  057442  000240                        NOP
12251  057444                          60$:
12252
12253                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
12254  057444  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
12255  057452  001432                        BEQ     70$             ;NO!!
```

```
12256                                   ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
12257  057454  013737  001372  001140           MOV     RMER21,$GDDAT   ;EXPECTED STATUS
12258  057462  042737  010000  001140           BIC     #IVC,$GDDAT
12259  057470  013737  001372  001142           MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12260  057476  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12261  057502  112776  000300  000000           MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
12262  057510  032737  000100  001342           BIT     #VV,RMDS1       ;WAS VOLUME VALID??
12263  057516  001403                           BEQ     65$             ;NO!!
12264  057520  112776  000301  000000           MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
12265  057526  162716  000002           65$:    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12266  057532  004736                           JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
12267  057534  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12268  057540                           70$:
12269
12270                                   ;SEE IF "ILF" OR "RMR" IS SET
12271  057540  032737  000007  001344           BIT     #ILR!ILF!RMR, RMER11
12272  057546  001510                           BEQ     100$            ;NO ERRORS DETECTED
12273                                   ;REPORT AN ERROR IF "ILR" IS SET
12274  057550  032737  000002  001344           BIT     #ILR,RMER11     ;WAS "ILR" DETECTED??
12275  057556  001424                           BEQ     80$             ;NO!!
12276  057560  013737  001344  001140           MOV     RMER11,$GDDAT   ;EXPECTED STATUS
12277  057566  042737  000002  001140           BIC     #ILR,$GDDAT
12278  057574  013737  001344  001142           MOV     RMER11,$BDDAT   ;RECEIVED STATUS
12279  057602  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12280  057606  112776  000302  000000           MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
12281  057614  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12282  057620  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12283  057622  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12284  057626  000240                           NOP
12285  057630                           80$:
12286
12287                                   ;REPORT AN ERROR IF "ILF" IS SET
12288  057630  032737  000001  001344           BIT     #ILF,RMER11     ;WAS "ILF" DETECTED??
12289  057636  001424                           BEQ     90$             ;NO!!
12290  057640  013737  001344  001140           MOV     RMER11,$GDDAT   ;EXPECTED STATUS
12291  057646  042737  000001  001140           BIC     #ILF,$GDDAT
12292  057654  013737  001344  001142           MOV     RMER11,$BDDAT   ;RECEIVED STATUS
12293  057662  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12294  057666  112776  000303  000000           MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
12295  057674  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12296  057700  004736                           JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
12297  057702  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12298  057706  000240                           NOP
12299  057710                           90$:
12300
12301                                   ;REPORT AN ERROR IF "RMR" IS SET
12302  057710  032737  000004  001344           BIT     #RMR,RMER11     ;WAS "RMR" DETECTED??
12303  057716  001424                           BEQ     100$            ;NO!!
12304  057720  013737  001344  001140           MOV     RMER11,$GDDAT   ;EXPECTED STATUS
12305  057726  042737  000004  001140           BIC     #RMR,$GDDAT
12306  057734  013737  001344  001142           MOV     RMER11,$BDDAT   ;RECEIVED STATUS
12307  057742  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12308  057746  112776  000304  000000           MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
12309  057754  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12310  057760  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12311  057762  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
12312  057766  000240                          NOP
12313  057770                          100$:
12314                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
12315  057770  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
12316  057776  052737  040000  062432          BIS     #SKI,500$       ;SET SKI FLAG
12317  060004  022737  001466  001434          CMP     #822.,RMDC0     ;IS CYLINDER > 822??
12318  060012  103425                          BLO     110$            ;YES!!
12319  060014  042737  040000  062432          BIC     #SKI,500$       ;RESET SKI FLAG
12320  060022  122737  000004  001407          CMPB    #4,RMDA0+1      ;IS TRACK > 4??
12321  060030  103416                          BLO     110$            ;YES!!
12322  060032  122737  000037  001406          CMPB    #31.,RMDA0      ;IS SECTOR > 31??
12323  060040  103412                          BLO     110$            ;YES!!
12324  060042  122737  000035  001406          CMPB    #29.,RMDA0      ;IS SECTOR > 29??
12325  060050  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
12326  060052  032737  010000  001432          BIT     #FMT16,RMOF0    ;18 BIT FORMAT??
12327  060060  001402                          BEQ     110$            ;YES!!
12328  060062  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
12329  060066  013737  001344  001142  110$:   MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
12330  060074  042737  175777  001142          BIC     #^CIAE,$BDDAT
12331  060102  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
12332  060110  001004                          BNE     115$            ;NO!!
12333  060112  042737  040000  062432          BIC     #SKI,500$       ;IAE OK - SKI SHOULD BE 0
12334  060120  000412                          BR      120$
12335  060122  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12336  060126  112776  000305  000000          MOVB    #305,@(SP)      ;WRITE ERROR NUMBER
12337  060134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12338  060140  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12339  060142  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12340  060146                          120$:
12341
12342                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
12343  060146  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12344  060154  042737  137777  001142          BIC     #^CSKI,$BDDAT
12345  060162  013737  062432  001140          MOV     500$,$GDDAT     ;EXPECTED STATUS
12346  060170  042737  137777  001140          BIC     #^CSKI,$GDDAT
12347  060176  032737  040000  001372          BIT     #SKI,RMER21     ;WAS "SKI" SET??
12348  060204  001417                          BEQ     140$            ;NO!!
12349  060206  032737  040000  062432          BIT     #SKI,500$       ;WAS SKI CAUSED BY IAE = 0??
12350  060214  001032                          BNE     150$            ;YES - DON'T REPORT SKI
12351  060216  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12352  060222  112776  000306  000000          MOVB    #306,@(SP)      ;WRITE ERROR NUMBER
12353  060230  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12354  060234  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12355  060236  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12356  060242  000417                          BR      150$
12357
12358  060244                          140$:
12359
12360                                  ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
12361  060244  032737  040000  062432          BIT     #SKI,500$       ;SHOULD SKI BE SET??
12362  060252  001413                          BEQ     150$            ;NO!!
12363  060254  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12364  060260  112776  000307  000000          MOVB    #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
12365  060266  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12366  060272  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12367  060274  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
12368  060300  000240                          NOP
12369  060302                          150$:
12370
12371                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
12372  060302  032737  006200  001372          BIT     #LSC!LBC!DVC,RMER21
12373  060310  001512                          BEQ     180$            ;NO ERRORS SET
12374
12375                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
12376  060312  032737  000200  001372          BIT     #DVC,RMER21     ;IS "DVC" = 1??
12377  060320  001424                          BEQ     160$            ;NO!!
12378  060322  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
12379  060330  042737  000200  001140          BIC     #DVC,$GDDAT
12380  060336  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12381  060344  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12382  060350  112776  000310  000000          MOVB    #310,@(SP)      ;WRITE ERROR NUMBER IN CALL
12383  060356  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12384  060362  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12385  060364  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12386  060370  000240                          NOP
12387  060372                          160$:
12388
12389                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
12390  060372  032737  002000  001372          BIT     #LBC,RMER21     ;IS LBC SET??
12391  060400  001430                          BEQ     170$            ;NO!!
12392  060402  032737  010000  001342          BIT     #MOL,RMDS1      ;WAS LBC ERROR BY MOL = 0
12393  060410  001424                          BEQ     170$            ;YES!!
12394  060412  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
12395  060420  042737  002000  001140          BIC     #LBC,$GDDAT
12396  060426  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12397  060434  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12398  060440  112776  000311  000000          MOVB    #311,@(SP)      ;WRITE ERROR NUMBER IN CALL
12399  060446  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12400  060452  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12401  060454  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12402  060460  000240                          NOP
12403  060462                          170$:
12404
12405                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
12406  060462  032737  004000  001372          BIT     #LSC,RMER21     ;IS "LSC" = 1??
12407  060470  001422                          BEQ     180$            ;NO!!
12408  060472  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
12409  060500  042737  004000  001140          BIC     #LSC,$GDDAT
12410  060506  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12411  060514  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12412  060520  112776  000312  000000          MOVB    #312,@(SP)      ;WRITE ERROR NUMBER
12413  060526  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12414  060530  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12415  060534  000240                          NOP
12416  060536                          180$:
12417
12418                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
12419  060536  032737  054000  001344          BIT     #UNS!DTE!WLE,RMER11
12420  060544  001527                          BEQ     220$            ;NO BITS SET
12421                                  ;REPORT "UNS" ERROR IF "DVC" = 0
12422  060546  032737  040000  001344          BIT     #UNS,RMER11     ;IS "UNS" SET??
12423  060554  001427                          BEQ     190$            ;NO!!
```

```
12424  060556  032737  000200  001372          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
12425  060564  001023                          BNE     190$            ;YES!!
12426  060566  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12427  060574  042737  040000  001140          BIC     #UNS,$GDDAT
12428  060602  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12429  060610  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12430  060614  112776  000313  000000          MOVB    #313,@(SP)      ;WRITE ERROR NUMBER
12431  060622  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12432  060626  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12433  060630  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12434  060634                          190$:
12435
12436                                  ;REPORT ANY DRIVE TIMING ERROR, I.E., "DTE" = 1
12437  060634  032737  010000  001344          BIT     #DTE,RMER1I     ;IS DTE SET??
12438  060642  001423                          BEQ     200$            ;NO!!
12439  060644  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12440  060652  042737  010000  001140          BIC     #DTE,$GDDAT
12441  060660  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12442  060666  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12443  060672  112776  000314  000000          MOVB    #314,@(SP)      ;WRITE ERROR NUMBER IN CALL
12444  060700  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12445  060704  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12446  060706  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12447  060712                          200$:
12448
12449                                  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
12450                                  ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
12451  060712  032737  004000  001344          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
12452  060720  001441                          BEQ     220$            ;NO!!
12453  060722  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12454  060730  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12455  060736  052737  004000  001140          BIS     #WLE,$GDDAT
12456  060744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12457  060750  112776  000315  000000          MOVB    #315,@(SP)      ;WRITE ERROR NUMBER IN CALL
12458  060756  032737  004000  001342          BIT     #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
12459  060764  001404                          BEQ     205$            ;NO!!
12460  060766  032737  000010  001400          BIT     #BIT3,RMCS1O    ;WAS COMMAND A WRITE??
12461  060774  001406                          BEQ     210$            ;YES!!
12462  060776  112776  000316  000000  205$:   MOVB    #316,@(SP)      ;CHANGE ERROR NUMBER
12463  061004  042737  004000  001140          BIC     #WLE,$GDDAT
12464  061012  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12465  061016  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12466  061020  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12467
12468  061024                          220$:
12469
12470                                  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
12471  061024  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12472  061030  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
12473  061034  001404                          BEQ     225$            ;NO - DO DATA CHECKS
12474  061036  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
12475  061042  000137  062044                  JMP     340$            ;SKIP DATA CHECKS
12476  061046  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
12477
12478                                  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
12479                                  ;IF HEADER COMPARE IS NOT INHIBITED
```

```
12480  061052  013737  001400  062434          MOV     RMCS10,510$     ;STRIP AND STORE FUNCTION CODE
12481  061060  042737  177700  062434          BIC     #^CFNCMSK,510$
12482  061066  022737  000063  062434          CMP     #WH!GO,510$     ;WAS FUNCTION WRITE HEADER & DATA??
12483  061074  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
12484  061076  032737  002000  001362          BIT     #HCI,RMOFI      ;WAS HCI SET??
12485  061104  001106                          BNE     250$            ;YES - SKIP HEADER CHECKS
12486
12487                                  ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
12488  061106  032737  000620  001344          BIT     #HCRC!FER!HCE,RMER1I
12489  061114  001533                          BEQ     270$            ;NO ERRORS SET
12490
12491                                  ;REPORT HEADER CRC ERROR IF SET
12492  061116  032737  000400  001344          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
12493  061124  001422                          BEQ     230$            ;NO!!
12494  061126  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12495  061134  042737  000400  001140          BIC     #HCRC,$GDDAT
12496  061142  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12497  061150  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12498  061154  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
12499  061162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12500  061166  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12501  061170  000501                          BR      260$
12502  061172                          230$:
12503
12504                                  ;REPORT FORMAT ERROR IF SET
12505  061172  032737  000020  001344          BIT     #FER,RMER1I     ;WAS "FER" SET??
12506  061200  001422                          BEQ     240$            ;NO!!
12507  061202  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12508  061210  042737  000020  001140          BIC     #FER,$GDDAT
12509  061216  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12510  061224  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12511  061230  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
12512  061236  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12513  061242  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12514  061244  000453                          BR      260$
12515  061246                          240$:
12516
12517                                  ;REPORT HEADER COMPARE ERROR IF SET
12518  061246  032737  000200  001344          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
12519  061254  001453                          BEQ     270$            ;NO!!
12520  061256  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12521  061264  042737  000200  001140          BIC     #HCE,$GDDAT
12522  061272  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12523  061300  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12524  061304  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
12525  061312  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12526  061316  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12527  061320  000425                          BR      260$
12528                                  ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
12529                                  ;       .COMMAND WAS WRITE HEADER AND DATA, OR
12530                                  ;       .HEADER COMPARE INHIBIT WAS SET
12531  061322  032737  000620  001344  250$:   BIT     #HCE!FER!HCRC,RMER1I
12532  061330  001425                          BEQ     270$            ;NO ERRORS WERE SET
12533  061332  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12534  061340  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
12535  061346  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
12536  061354  062716  000004                ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
12537  061360  112776  000322  000000        MOVB    #322,@(SP)       ;WRITE ERROR NUMBER
12538  061366  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12539  061372  004736                        JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12540  061374  162716  000010        260$:   SUB     #10,(SP)         ;MOVE SP TO NO ERROR
12541  061400  000137  062044                JMP     340$             ;OMIT FURTHER DATA CHECKS
12542
12543  061404                        270$:
12544
12545                                ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
12546                                ;DO READ ERROR CHECKS
12547  061404  032737  000010  062434        BIT     #BIT3,510$       ;WAS THIS A WRITE COMMAND?
12548  061412  001002                        BNE     275$             ;NO!!
12549  061414  000137  061632                JMP     310$             ;GO DO WRITE STATUS CHECK
12550  061420                        275$:
12551
12552                                ;REPORT DATA CHECK IF SET
12553  061420  032737  100000  001344        BIT     #DCK,RMER11      ;DATA CHECK ERROR??
12554  061426  001450                        BEQ     290$             ;NO!!
12555  061430  013737  001344  001140        MOV     RMER11,$GDDAT    ;EXPECTED STATUS
12556  061436  042737  100000  001140        BIC     #DCK,$GDDAT
12557  061444  013737  001344  001142        MOV     RMER11,$BDDAT    ;RECEIVED STATUS
12558  061452  062716  000004                ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
12559  061456  112776  000323  000000        MOVB    #323,@(SP)       ;WRITE ERROR NUMBER
12560  061464  032737  004000  001362        BIT     #ECI,RMOFI       ;WAS ECC CORRECTION DISABLED??
12561  061472  001021                        BNE     280$             ;YES!!
12562  061474  112776  000324  000000        MOVB    #324,@(SP)       ;CHANGE TO RECOVERABLE ERROR
12563  061502  032737  000100  001344        BIT     #ECH,RMER11      ;IS ERROR RECOVERABLE??
12564  061510  001007                        BNE     276$             ;NO !!
12565                                ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
12566  061512  032737  000020  062434        BIT     #BIT4,510$       ;WAS THIS A READ COMMAND ??
12567  061520  001406                        BEQ     280$             ;NO !!
12568  061522  162716  000004                SUB     #4,(SP)          ;RESTORE SP
12569  061526  000410                        BR      290$             ;SKIP ERROR - DATA WILL BE CORRECTED
12570  061530  112776  000325  000000  276$: MOVB    #325,@(SP)       ;CHANGE TO NON RECOVERABLE
12571  061536  162716  000002        280$:   SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12572  061542  004736                        JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12573  061544  162716  000010                SUB     #10,(SP)         ;RESTORE SP TO NO ERROR
12574
12575  061550                        290$:
12576
12577                                ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
12578  061550  032737  000400  001340        BIT     #MDPE,RMCS21     ;PARITY ERROR SET??
12579  061556  001423                        BEQ     300$             ;NO!!
12580  061560  013737  001340  001140        MOV     RMCS21,$GDDAT    ;EXPECTED STATUS
12581  061566  042737  000400  001140        BIC     #MDPE,$GDDAT
12582  061574  013737  001340  001142        MOV     RMCS21,$BDDAT    ;RECEIVED STATUS
12583  061602  062716  000004                ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
12584  061606  112776  000326  000000        MOVB    #326,@(SP)       ;WRITE ERROR NUMBER
12585  061614  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12586  061620  004736                        JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12587  061622  162716  000010                SUB     #10,(SP)         ;MOVE SP TO NO ERROR
12588  061626  000137  062044        300$:   JMP     340$             ;SKIP WRITE STATUS CHECK
12589
12590  061632                        310$:
12591
```

```
12592                                   ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
12593  061632  032737  000001  001342            BIT     #OM,RMDSI       ;IS OFFSET ON??
12594  061640  001423                            BEQ     320$            ;NO
12595  061642  013737  001342  001140            MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12596  061650  042737  000001  001140            BIC     #OM,$GDDAT
12597  061656  013737  001342  001142            MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12598  061664  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12599  061670  112776  000327  000000            MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
12600  061676  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12601  061702  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12602  061704  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12603  061710                          320$:
12604
12605                                   ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
12606  061710  032737  000010  001372            BIT     #DPE,RMER2I     ;DATA PARITY ERROR??
12607  061716  001423                            BEQ     330$            ;NO!!
12608  061720  013737  001372  001140            MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12609  061726  042737  000010  001140            BIC     #DPE,$GDDAT
12610  061734  013737  001372  001142            MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12611  061742  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12612  061746  112776  000330  000000            MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
12613  061754  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12614  061760  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12615  061762  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12616  061766                          330$:
12617
12618                                   ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
12619  061766  032737  000040  001344            BIT     #WCF,RMER1I     ;IS "WCF" SET??
12620  061774  001423                            BEQ     340$            ;NO!!
12621  061776  013737  001344  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12622  062004  042737  000040  001140            BIC     #WCF,$GDDAT
12623  062012  013737  001344  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12624  062020  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12625  062024  112776  000331  000000            MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
12626  062032  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12627  062036  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12628  062040  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12629  062044                          340$:
12630
12631                                   ;REPORT "DATA LATE" ERROR IF "DLT" = 1
12632  062044  032737  100000  001340            BIT     #DLT,RMCS2I     ;IS "DLT" SET??
12633  062052  001423                            BEQ     350$            ;NO!!
12634  062054  013737  001340  001140            MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12635  062062  042737  100000  001140            BIC     #DLT,$GDDAT
12636  062070  013737  001340  001142            MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12637  062076  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12638  062102  112776  000332  000000            MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
12639  062110  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12640  062114  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12641  062116  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12642  062122                          350$:
12643                                   ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
12644  062122  013746  001342                    MOV     RMDSI,-(SP)     ;STACK DRIVE STATUS
12645  062126  042716  147677                    BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
12646  062132  022726  010100                    CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
12647  062136  001522                            BEQ     380$            ;YES!!
```

```
12648
12649                                   ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
12650                                   ;I.E, PIP = 1 AND SKI = 0
12651  062140  032737  020000  001342          BIT     #PIP,RMDSI      ;IS "PIP" SET??
12652  062146  001430                        BEQ     360$            ;NO!!
12653  062150  032737  040000  001372        BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
12654  062156  001024                        BNE     360$            ;YES-DONT REPORT PIP
12655  062160  013737  001342  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12656  062166  042737  020000  001140        BIC     #PIP,$GDDAT
12657  062174  013737  001342  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12658  062202  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12659  062206  112776  000333  000000        MOVB    #333,@(SP)      ;WRITE ERROR NUMBER
12660  062214  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12661  062220  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12662  062222  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12663  062226  000240                        NOP
12664  062230                          360$:
12665
12666                                   ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
12667                                   ;REPORTED, I.E., MOL = OPI = 0
12668  062230  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
12669  062236  001027                        BNE     370$            ;YES!!
12670  062240  032737  020000  001344        BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
12671  062246  001023                        BNE     370$            ;YES!!
12672  062250  013737  001342  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12673  062256  052737  010000  001140        BIS     #MOL,$GDDAT
12674  062264  013737  001342  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12675  062272  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12676  062276  112776  000334  000000        MOVB    #334,@(SP)      ;WRITE ERROR NUMBER
12677  062304  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12678  062310  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12679  062312  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12680  062316                          370$:
12681
12682                                   ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
12683                                   ;REPORTED, I.E., VV = IVC = 0
12684  062316  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID??
12685  062324  001027                        BNE     380$            ;YES!!
12686  062326  032737  010000  001372        BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
12687  062334  001033                        BNE     390$            ;YES!!
12688  062336  013737  001342  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12689  062344  052737  000100  001140        BIS     #VV,$GDDAT
12690  062352  013737  001342  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12691  062360  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12692  062364  112776  000335  000000        MOVB    #335,@(SP)      ;WRITE ERROR NUMBER
12693  062372  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12694  062376  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12695  062400  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12696  062404                          380$:
12697
12698                                   ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
12699  062404  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12700  062410  105776  000000                TSTB    @(SP)           ;ANY ERROR??
12701  062414  001403                        BEQ     390$            ;NO!!
12702  062416  062716  000004                ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12703  062422  000402                        BR      400$
```

```
12704  062424  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12705
12706  062430  000207                  400$:   RTS     PC              ;RETURN TO USER
12707
12708  062432  000000                  500$:   .WORD                   ;ERROR FLAGS
12709  062434  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
12710                                   .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
12711
12712                                   ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
12713                                   ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
12714                                   ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
12715                                   ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
12716
12717                                   ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
12718                                   ;IF TRUE:
12719
12720                                   ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
12721                                   ;THAT MOL IS ASSUMED TO HAVE BEEN SET
12722                                   ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
12723                                   ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
12724                                   ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
12725                                   ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
12726
12727                                   ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
12728
12729                                   ;(1)    JSR     PC,STCDRVSTS
12730                                   ;       BR      ???             RETURN HERE IF NO ERROR
12731                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12732                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12733                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12734                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12735
12736  062436                           STCDRVSTS:
12737
12738                                   ;CLEAR USER'S ERROR CALL
12739  062436  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12740  062442  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
12741  062446  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
12742                                   ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
12743  062452  013746  001342                  MOV     RMDS1,-(SP)     ;PUT DRIVE STATUS ON STACK
12744  062456  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
12745  062462  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
12746  062466  001524                          BEQ     30$             ;YES!!
12747
12748                                   ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
12749  062470  032737  010000  001342          BIT     #MOL,RMDS1      ;IS MOL ON ??
12750  062476  001030                          BNE     10$             ;YES!!
12751  062500  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
12752  062506  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
12753  062510  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
12754  062516  052737  010000  001140          BIS     #MOL,$GDDAT
12755  062524  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
12756  062532  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12757  062536  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
12758  062544  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12759  062550  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12760  062552  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
12761  062556  000240                          NOP
12762  062560                          10$:
12763
12764                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
12765  062560  032737  000100  001342          BIT     #VV,RMDS1       ;IS "VV" = 0??
12766  062566  001030                          BNE     20$             ;NO!!
12767  062570  032737  010000  001372          BIT     #IVC,RMER21     ;WAS "IVC" SET??
12768  062576  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
12769  062600  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
12770  062606  052737  000100  001342          BIS     #VV,RMDS1
12771  062614  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
12772  062622  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12773  062626  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
12774  062634  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12775  062640  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12776  062642  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12777  062646  000240                          NOP
12778  062650                          20$:
12779
12780                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
12781  062650  032737  020000  001342          BIT     #PIP,RMDS1      ;IS DRIVE OFF CYLINDER??
12782  062656  001430                          BEQ     30$             ;NO!!
12783  062660  032737  040000  001372          BIT     #SKI,RMER21     ;WAS "SKI" SET??
12784  062666  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
12785  062670  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
12786  062676  042737  020000  001140          BIC     #PIP,$GDDAT
12787  062704  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
12788  062712  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12789  062716  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
12790  062724  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12791  062730  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12792  062732  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12793  062736  000240                          NOP
12794  062740                          30$:
12795
12796                                  ;SEE IF "SKI" = "DVC" = 0
12797  062740  013746  001372                  MOV     RMER21,-(SP)    ;PUT ERROR REG 2 ON STACK
12798  062744  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
12799  062750  001460                          BEQ     60$             ;BRANCH IF NO ERROR
12800  062752                          40$:
12801
12802                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
12803  062752  032737  000200  001372          BIT     #DVC,RMER21     ;ANY DEVICE FAULT??
12804  062760  001424                          BEQ     50$             ;NO!!
12805  062762  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
12806  062770  042737  000200  001140          BIC     #DVC,$GDDAT
12807  062776  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
12808  063004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
12809  063010  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12810  063016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12811  063022  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12812  063024  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12813  063030  000240                          NOP
12814  063032                          50$:
12815
12816                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
12817  063032  032737  040000  001372          BIT     #SK1,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
12818  063040  001424                          BEQ     60$             ;NO!!
12819  063042  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12820  063050  042737  040000  001140          BIC     #SK1,$GDDAT
12821  063056  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12822  063064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12823  063070  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
12824  063076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12825  063102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12826  063104  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12827  063110  000240                          NOP
12828  063112                          60$:
12829
12830                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12831  063112  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12832  063116  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
12833  063122  001403                          BEQ     70$             ;NO!!
12834  063124  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
12835  063130  000402                          BR      80$
12836  063132  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
12837  063136  000240                  80$:    NOP
12838  063140  000207                          RTS     PC              ;RETURN TO USER
```

```
12839                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
12840
12841                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
12842                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
12843                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
12844                                   ;THE MASKS ARE APPLIED.
12845
12846                                   ;CALL:
12847                                   ;(1)    JSR     PC,CMPERRSTS
12848                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
12849                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
12850                                   ;       BR      ???             RETURN HERE IF NO ERROR
12851                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12852                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12853                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12854                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12855
12856                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
12857                                   ;BE ZERO
12858
12859  063142                           CMPERRSTS:
12860
12861                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
12862  063142  013737  001344  001176           MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
12863  063150  047637  000000  001176           BIC     @(SP),$TMP1     ;MASK   RMER1
12864  063156  062716  000002                   ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
12865  063162  013737  001372  001200           MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
12866  063170  047637  000000  001200           BIC     @(SP),$TMP2     ;MASK RMER2
12867
12868
12869                                   ;CLEAR USER'S ERROR CALL
12870  063176  062716  000006                   ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
12871  063202  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
12872  063206  162716  000004                   SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
12873
12874                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
12875  063212  005737  001176                   TST     $TMP1           ;ANY ERRORS TO REPORT??
12876  063216  001420                           BEQ     5$              ;NO !!
12877  063220  013737  001176  001142           MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12878  063226  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12879  063232  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12880  063236  112776  000066  000000           MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
12881  063244  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12882  063250  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12883  063252  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
12884  063256  000240                           NOP
12885  063260                           5$:
12886
12887
12888                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
12889  063260  005737  001200                   TST     $TMP2           ;ANY ERRORS IN RMER2?
12890  063264  001420                           BEQ     10$             ;NO!!
12891
12892  063266  013737  001200  001142           MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12893  063274  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12894  063300  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
12895  063304  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
12896  063312  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12897  063316  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12898  063320  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12899  063324  000240                          NOP
12900  063326                          10$:
12901
12902                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12903  063326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12904  063332  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
12905  063336  001403                          BEQ     20$             ;NO!!
12906  063340  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12907  063344  000402                          BR      30$
12908  063346  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12909  063352  000207                  30$:    RTS     PC              ;RETURN TO USER
12910
12911
```

```
12912                                   .SBTTL  STOP AND SHUTDOWN SUBROUTINES
12913
12914  063354                           STOP:
12915
12916                                   ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
12917  063354  012746  000140                 MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
12918  063360  012746  063366                 MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12919  063364  000002                         RTI                     ;;POP NEW PC AND PS
12920  063366                           64$:
12921  063366  000240                         NOP
12922                                   ;RAISE PRIORITY TO INHIBIT INTERRUPT
12923  063370  012746  000300                 MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
12924  063374  012746  063402                 MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
12925  063400  000002                         RTI                     ;;POP NEW PC AND PS
12926  063402                           65$:
12927
12928  063402  000207                 10$:    RTS     PC              ;CONTINUE
12929
12930
12931  063404  012737  177777  001326 SHUT2:  MOV     #-1,CTLFG       ;SET CONTROL-C FLAG             ;;**C
12932  063412                           SHUT:
12933  063412  005737  001326                 TST     @#CTLFG         ;HALT ?
12934  063416  001002                         BNE     .+6             ;BRANCH IF SO
12935  063420  000137  007100                 JMP     READY           ;OTHERWISE LOOP
12936  063424  104401  063446                 TYPE    ,100$           ;TYPE THE HALT MESSAGE
12937  063430  005737  000042                 TST     42              ;RUNNING STANDALONE ??
12938  063434  001402                         BEQ     10$             ;YES !!
12939  063436  000137  037006                 JMP     $EOP            ;NO - GO TO END OF PASS
12940  063442                           10$:
12941  063442  000137  005324                 JMP     START           ;GO TO START
12942  063446  042524  052123  053440 100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
12943  063454  051501  044040  046101
12944  063462  042524  006504  005012
12945  063470     000
12946          063472                   .EVEN
```

```
12947                                   .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
12948
12949                                   ;;**********************************************************************
12950                                   ;*SAVE R0-R5
12951                                   ;*CALL:
12952                                   ;*      SAVREG
12953                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
12954                                   ;*
12955                                   ;*TOP---(+16)
12956                                   ;* +2---(+18)
12957                                   ;* +4---R5
12958                                   ;* +6---R4
12959                                   ;* +8---R3
12960                                   ;*+10---R2
12961                                   ;*+12---R1
12962                                   ;*+14---R0
12963
12964  063472                           $SAVREG:
12965  063472  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12966  063474  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12967  063476  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12968  063500  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12969  063502  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12970  063504  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12971  063506  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
12972  063512  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
12973  063516  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
12974  063522  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
12975  063526  000002                          RTI
12976
12977                                   ;*RESTORE R0-R5
12978                                   ;*CALL:
12979                                   ;*      RESREG
12980  063530                           $RESREG:
12981  063530  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
12982  063534  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
12983  063540  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
12984  063544  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
12985  063550  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12986  063552  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12987  063554  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12988  063556  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12989  063560  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12990  063562  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12991  063564  000002                          RTI
12992                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
12993
12994                                   ;;**********************************************************************
12995                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
12996                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
12997                                   ;*CALL:
12998                                   ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
12999                                   ;*      TYPBN                   ;;TYPE IT
13000
13001  063566  010146                   $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
13002  063570  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
13003  063574  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
13004  063576  112737  000060  063640  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
13005  063604  006101                          ROL     R1              ;;GET THIS BIT
13006  063606  001406                          BEQ     2$              ;;DONE?
13007  063610  105537  063640                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
13008  063614  104401  063640                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
13009  063620  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
13010  063622  000765                          BR      1$              ;;GO DO THE NEXT BIT
13011  063624  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
13012  063626  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
13013  063634  012616                          MOV     (SP)+,(SP)
13014  063636  000002                          RTI                     ;;RETURN TO USER
13015  063640     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
13016                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
13017
13018                                  ;;**********************************************************************
13019                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
13020                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
13021                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
13022                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
13023                                  ;*REPLACED WITH SPACES.
13024                                  ;*CALL:
13025                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
13026                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
13027
13028  063642                          $TYPDS:
13029  063642  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13030  063644  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13031  063646  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13032  063650  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
13033  063652  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
13034  063654  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
13035  063660  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
13036  063664  100004                          BPL     1$              ;;BR IF INPUT IS POS.
13037  063666  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
13038  063670  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
13039  063676  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
13040  063700  012703  064056                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
13041  063704  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
13042  063710  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
13043  063712  016001  064046                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
13044  063716  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
13045  063720  002402                          BLT     4$              ;;BR IF DONE
13046  063722  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
13047  063724  000774                          BR      3$
13048  063726  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
13049  063730  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
13050  063732  001002                          BNE     5$              ;;FALL THROUGH IF 0
13051  063734  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
13052  063736  100407                          BMI     7$              ;;BR IF YES
13053  063740  106316                  5$:     ASLB    (SP)            ;;MSD?
13054  063742  103003                          BCC     6$              ;;BR IF NO
13055  063744  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
13056  063752  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
13057  063756  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
13058  063762  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
13059  063764  005720                          TST     (R0)+           ;;JUST INCREMENTING
13060  063766  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
13061  063772  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
13062  063774  003002                          BGT     8$              ;;GO TO EXIT
13063  063776  010502                          MOV     R5,R2           ;;GET THE LSD
13064  064000  000764                          BR      6$              ;;GO CHANGE TO ASCII
13065  064002  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
13066  064004  100003                          BPL     9$              ;;BR IF NO
13067  064006  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
13068  064014  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
13069  064016  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
13070  064020  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
13071  064022  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
13072  064024  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
13073  064026  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
13074  064030  104401  064056                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
13075  064034  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
13076  064042  012616                          MOV     (SP)+,(SP)
13077  064044  000002                          RTI                     ;;RETURN TO USER
13078  064046  023420                  $DTBL:  10000.
13079  064050  001750                          1000.
13080  064052  000144                          100.
13081  064054  000012                          10.
13082  064056  000004                  $DBLK:  .BLKW   4
13083                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
13084
13085                                  ;;**************************************************************
13086                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
13087                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
13088                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
13089                                  ;*CALL:
13090                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13091                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
13092                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
13093                                  ;*      .BYTE   M               ;;M=1 OR 0
13094                                  ;*                                      ;;1=TYPE LEADING ZEROS
13095                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
13096                                  ;*
13097                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
13098                                  ;*$TYPOS OR $TYPOC
13099                                  ;*CALL:
13100                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13101                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
13102                                  ;*
13103                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
13104                                  ;*CALL:
13105                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13106                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
13107
13108  064066  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
13109  064072  116637  000001  064311          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
13110  064100  112637  064313                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
13111  064104  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
13112  064110  000406                          BR      $TYPON
13113  064112  112737  000001  064311  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
13114  064120  112737  000006  064313          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
13115  064126  112737  000005  064310  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
13116  064134  010346                          MOV     R3,-(SP)        ;;SAVE R3
13117  064136  010446                          MOV     R4,-(SP)        ;;SAVE R4
13118  064140  010546                          MOV     R5,-(SP)        ;;SAVE R5
13119  064142  113704  064313                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
13120  064146  005404                          NEG     R4
13121  064150  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
13122  064154  110437  064312                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
13123  064160  113704  064311                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
13124  064164  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
13125  064170  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
13126  064172  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
13127  064174  000404                          BR      3$              ;;GO DO MSB
13128  064176  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
13129  064200  006105                          ROL     R5
13130  064202  006105                          ROL     R5
13131  064204  010503                          MOV     R5,R3
13132  064206  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
13133  064210  105337  064312                  DECB    $OMODE          ;;TYPE THIS DIGIT?
13134  064214  100016                          BPL     7$              ;;BR IF NO
13135  064216  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
13136  064222  001002                          BNE     4$              ;;TEST FOR 0
13137  064224  005704                          TST     R4              ;;SUPPRESS THIS 0?
13138  064226  001403                          BEQ     5$              ;;BR IF YES
13139  064230  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
13140  064232  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
13141  064236  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
13142  064242  110337  064306                  MOVB    R3,8$           ;;SAVE FOR TYPING
13143  064246  104401  064306                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
13144  064252  105337  064310          7$:     DECB    $OCNT           ;;COUNT BY 1
13145  064256  003347                          BGT     2$              ;;BR IF MORE TO DO
13146  064260  002402                          BLT     6$              ;;BR IF DONE
13147  064262  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
13148  064264  000744                          BR      2$              ;;GO DO THE LAST DIGIT
13149  064266  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
13150  064270  012604                          MOV     (SP)+,R4        ;;RESTORE R4
13151  064272  012603                          MOV     (SP)+,R3        ;;RESTORE R3
13152  064274  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
13153  064302  012616                          MOV     (SP)+,(SP)
13154  064304  000002                          RTI                     ;;RETURN
13155  064306     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
13156  064307     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
13157  064310     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
13158  064311     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
13159  064312  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
13160                                  .SBTTL  TYPE ROUTINE
13161
13162                                  ;;**************************************************************
13163                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13164                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13165                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13166                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13167                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13168                                  ;*
13169                                  ;*CALL:
13170                                  ;*1) USING A TRAP INSTRUCTION
```

```
13171                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13172                                   ;*OR
13173                                   ;*      TYPE
13174                                   ;*      MESADR
13175                                   ;*
13176
13177  064314  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
13178  064320  100002                          BPL     1$              ;;BR IF YES
13179  064322  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
13180  064324  000430                          BR      3$              ;;LEAVE
13181  064326  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
13182  064330  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
13183  064334  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13184  064342  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
13185  064344  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
13186  064352  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
13187  064354  010037  064364                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
13188  064360  004737  067224                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
13189  064364  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
13190  064366  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
13191  064374  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
13192  064376  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
13193  064400  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
13194  064402  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
13195  064404  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
13196  064406  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
13197  064412  000002                          RTI                     ;;RETURN
13198  064414  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
13199  064420  001430                          BEQ     8$
13200  064422  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
13201  064426  001006                          BNE     5$
13202  064430  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
13203  064432  104401                          TYPE                    ;;TYPE A CR AND LF
13204  064434  001217                          $CRLF
13205  064436  105037  064572                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
13206  064442  000755                          BR      2$              ;;GET NEXT CHARACTER
13207  064444  004737  064526          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
13208  064450  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
13209  064454  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
13210  064456  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
13211                                                                  ;;AND THE NULL CHAR.
13212  064462  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
13213  064466  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
13214  064470  004737  064526                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
13215  064474  105337  064572                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
13216  064500  000770                          BR      7$              ;;LOOP
13217
13218                                   ;HORIZONTAL TAB PROCESSOR
13219
13220  064502  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
13221  064506  004737  064526          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
13222  064512  132737  000007  064572          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
13223  064520  001372                          BNE     9$              ;;TAB STOP
13224  064522  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
13225  064524  000724                          BR      2$              ;;GET NEXT CHARACTER
13226  064526  105777  114432          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
13227  064532  100375                          BPL     $TYPEC
13228  064534  116677  000002  114424          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13229  064542  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
13230  064550  001003                          BNE     1$              ;;BRANCH IF NO
13231  064552  105037  064572                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
13232  064556  000406                          BR      $TYPEX          ;;EXIT
13233  064560  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
13234  064566  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
13235  064570  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
13236  064572  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
13237  064574  000207                  $TYPEX: RTS     PC
13238
13239                                  .SBTTL  SCOPE HANDLER ROUTINE
13240
13241                                  ;;**************************************************************
13242                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
13243                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
13244                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
13245                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13246                                  ;*SW14=1        LOOP ON TEST
13247                                  ;*SW11=1        INHIBIT ITERATIONS
13248                                  ;*SW09=1        LOOP ON ERROR
13249                                  ;*SW08=1        LOOP ON TEST IN SWR<7:0>
13250                                  ;*CALL
13251                                  ;*      SCOPE           ;;SCOPE=IOT
13252
13253  064576                          $SCOPE:
13254  064576  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
13255  064600  004737  063354                  JSR     PC,STOP
13256  064604  032777  040000  114342  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
13257  064612  001131                          BNE     $OVER           ;;YES IF SW14=1
13258                                  ;#####START OF CODE FOR THE XOR TESTER#####
13259  064614  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
13260                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
13261  064616  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
13262  064622  012737  064642  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
13263  064630  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
13264  064634  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
13265  064640  000500                          BR      $SVLAD          ;;GO TO THE NEXT TEST
13266  064642  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
13267  064644  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
13268  064650  000440                          BR      7$              ;;LOOP ON THE PRESENT TEST
13269  064652                          6$:;#####END OF CODE FOR THE XOR TESTER#####
13270  064652  032777  000400  114274          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
13271  064660  001421                          BEQ     2$              ;;BR IF NO
13272  064662  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
13273  064664  117716  114264                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
13274  064670  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
13275  064672  022716  000027                  CMP     #27,(SP)        ;;CHECK THE NUMBER IN THE SWR
13276  064676  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
13277  064700  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
13278  064704  005316                          DEC     (SP)            ;;BACKUP BY ONE
13279  064706  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
13280  064710  062716  065114                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
13281  064714  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
13282  064720  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
```

```
13283  064722  005726                    8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
13284  064724  105737  001117            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
13285  064730  001421                            BEQ     3$              ;;BR IF NO
13286  064732  123737  001131  001117            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
13287  064740  101015                            BHI     3$              ;;BR IF NO
13288  064742  032777  001000  114204            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
13289  064750  001404                            BEQ     4$              ;;BR IF NO
13290  064752  013737  001124  001122  7$:       MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
13291  064760  000446                            BR      $OVER
13292  064762  105037  001117            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
13293  064766  005037  001206                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
13294  064772  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
13295  064774  032777  004000  114152  3$:       BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
13296  065002  001011                            BNE     1$              ;;BR IF YES
13297  065004  005737  001230                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
13298  065010  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
13299  065012  005237  001120                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
13300  065016  023737  001206  001120            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
13301  065024  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
13302  065026  012737  000001  001120  1$:       MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
13303  065034  013737  065112  001206            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
13304  065042  105237  001116            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
13305  065046  113737  001116  001226            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
13306  065054  011637  001122                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
13307  065060  011637  001124                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
13308  065064  005037  001210                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
13309  065070  112737  000001  001131            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
13310  065076  013777  001116  114052  $OVER:    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
13311  065104  013716  001122                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
13312  065110  000002                            RTI                     ;;FIXES PS
13313  065112  000012                    $MXCNT: 10.                     ;;MAX. NUMBER OF ITERATIONS
13314  065114                            $SW08TBL:
13315  065114  007140                            .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
13316  065116  007336                            .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
13317  065120  007522                            .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
13318  065122  007704                            .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
13319  065124  010726                            .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
13320  065126  011720                            .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
13321  065130  013160                            .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
13322  065132  014202                            .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
13323  065134  015174                            .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
13324  065136  016436                            .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
13325  065140  017456                            .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
13326  065142  021004                            .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
13327  065144  022024                            .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
13328  065146  023044                            .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
13329  065150  024742                            .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
13330  065152  026416                            .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
13331  065154  027524                            .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
13332  065156  030472                            .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
13333  065160  031550                            .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
13334  065162  032636                            .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
13335  065164  033630                            .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
13336  065166  034652                            .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
13337  065170  035674                            .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
13338                            .SBTTL  ERROR HANDLER ROUTINE
```

```
13339
13340                                   ;;*****************************************************************
13341                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
13342                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
13343                                   ;*AND GO TO ERRTYP ON ERROR
13344                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13345                                   ;*SW15=1        HALT ON ERROR
13346                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
13347                                   ;*SW10=1        BELL ON ERROR
13348                                   ;*SW09=1        LOOP ON ERROR
13349                                   ;*CALL
13350                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
13351
13352  065172                           $ERROR:
13353  065172  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13354  065174  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
13355  065200  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
13356  065202  013777  001116  113746        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
13357  065210  032777  002000  113736        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
13358  065216  001402                        BEQ     1$              ;;NO - SKIP
13359  065220  104401  001212                TYPE    ,$BELL          ;;RING BELL
13360  065224  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
13361  065230  011637  001132                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
13362  065234  162737  000002  001132        SUB     #2,$ERRPC
13363  065242  117737  113664  001130        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
13364  065250  032777  020000  113676        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
13365  065256  001004                        BNE     20$             ;;SKIP TYPEOUTS
13366  065260  004737  037216                JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
13367  065264  104401  001217                TYPE    ,$CRLF
13368  065270                           20$:
13369  065270  122737  000001  001242        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13370  065276  001007                        BNE     2$              ;;NO,SKIP APT ERROR REPORT
13371  065300  113737  001130  065312        MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
13372  065306  004737  067234                JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
13373  065312     000                   21$:    .BYTE   0
13374  065313     000                           .BYTE   0
13375  065314  000777                  22$:    BR      22$             ;;APT ERROR LOOP
13376  065316  005777  113632          2$:     TST     @SWR            ;;HALT ON ERROR
13377  065322  100002                        BPL     3$              ;;SKIP IF CONTINUE
13378  065324  000000                        HALT                    ;;HALT ON ERROR!
13379  065326  104410                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13380  065330  032777  001000  113616  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
13381  065336  001402                        BEQ     4$              ;;BR IF NO
13382  065340  013716  001124                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
13383  065344  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
13384  065350  001402                        BEQ     5$              ;;BR IF NONE
13385  065352  013716  001210                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
13386  065356                           5$:
13387  065356  022737  037176  000042        CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
13388  065364  001001                        BNE     6$              ;;BRANCH IF NO
13389  065366  000000                        HALT                    ;;YES
13390  065370                           6$:
13391  065370  000002                        RTI                     ;;RETURN
13392                                   .SBTTL  TTY INPUT ROUTINE
13393
13394                                   ;;*****************************************************************
```

```
13395                                   .ENABL  LSB
13396  065372  000000                   $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
13397  065374  000000                   $TKQIN: .WORD   0               ;;INPUT POINTER
13398  065376  000000                   $TKQOUT: .WORD  0               ;;OUTPUT POINTER
13399  065400  000001                   $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
13400          065401                   $TKQEND=.
13401          065402                   .EVEN
13402
13403                                   ;*TK INITIALIZE ROUTINE
13404                                   ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
13405                                   ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
13406                                   ;
13407                                   ;*CALL:
13408                                   ;*      JSR     PC,$TKINT
13409                                   ;*      RETURN
13410                                   ;
13411  065402  005037  065372           $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
13412  065406  012737  065400  065374           MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
13413  065414  013737  065374  065376           MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
13414  065422  012737  065452  000060           MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
13415  065430  012737  000200  000062           MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
13416  065436  005777  113520                   TST     @$TKB           ;;CLEAR DONE FLAG
13417  065442  012777  000100  113510           MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
13418  065450  000207                           RTS     PC              ;;RETURN TO CALLER
13419
13420                                   ;*TK SERVICE ROUTINE
13421                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
13422                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
13423                                   ;*IT IN THE QUEUE.
13424                                   ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
13425                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT2)
13426                                   ;
13427  065452  117746  113504           $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
13428  065456  042716  177600                   BIC     #^C177,(SP)     ;;STRIP THE JUNK
13429  065462  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
13430  065466  001007                           BNE     1$              ;;BRANCH IF NO
13431  065470  104401  066566                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
13432  065474  004737  065402                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
13433  065500  005726                           TST     (SP)+           ;;CLEAN UP STACK
13434  065502  000137  063404                   JMP     SHUT2           ;;CONTROL C RESTART
13435  065506  021627  000007           1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
13436  065512  001004                           BNE     2$              ;;BRANCH IF NO
13437  065514  022737  000176  001154           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
13438  065522  001500                           BEQ     6$              ;;GO TO SWR CHANGE
13439
13440  065524                           2$:
13441  065524  022737  000001  065372           CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
13442  065532  001004                           BNE     3$              ;;BRANCH IF NO
13443  065534  104401  001212                   TYPE    ,$BELL          ;;RING THE TTY BELL
13444  065540  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
13445  065542  000451                           BR      5$              ;;EXIT
13446  065544  021627  000023           3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
13447  065550  001021                           BNE     32$             ;;BRANCH IF NO
13448  065552  005077  113402                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
13449  065556  005726                           TST     (SP)+           ;;CLEAN CHAR OFF STACK
13450  065560  105777  113374           31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
```

```
13451  065564  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
13452  065566  117746  113370                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
13453  065572  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13454  065576  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
13455  065602  001366                          BNE     31$             ;;BRANCH IF NO
13456  065604  012777  000100  113346          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
13457  065612  000002                          RTI                     ;;RETURN
13458  065614  005237  065372          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
13459  065620  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
13460  065624  002405                          BLT     4$              ;;BRANCH IF YES
13461  065626  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
13462  065632  003002                          BGT     4$              ;;BRANCH IF YES
13463  065634  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
13464  065640  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
13465  065644  005237  065374                  INC     $TKQIN          ;;UPDATE THE POINTER
13466  065650  023727  065374  065401          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
13467  065656  001003                          BNE     5$              ;;BRANCH IF NO
13468  065660  012737  065400  065374          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
13469  065666  000002                  5$:     RTI                     ;;RETURN
13470
13471                                  ;;**********************************************************************
13472                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
13473                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
13474                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
13475                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
13476  065670  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
13477  065676  001124                          BNE     15$             ;;EXIT IF NOT
13478  065700  105777  113254                  TSTB    @$TKS           ;;IS A CHAR WAITING?
13479  065704  100121                          BPL     15$             ;;IF NOT, EXIT
13480  065706  117746  113250                  MOVB    @$TKB,-(SP)     ;;YES
13481  065712  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
13482  065716  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
13483  065722  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
13484                                                                  ;;AND EXIT
13485
13486                                  ;;**********************************************************************
13487                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
13488                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
13489                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
13490  065724  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
13491  065732  001674                          BEQ     2$              ;;BRANCH IF YES
13492  065734  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
13493  065736  004737  065402                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
13494  065742  005077  113212                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
13495  065746  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
13496
13497  065754  104401  066600                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
13498  065760  104401  066605          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
13499  065764  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
13500  065770  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13501  065772  104401  066616                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
13502  065776  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
13503  066000  005046                          CLR     -(SP)           ;;THE NEW SWR
13504  066002  105777  113152          7$:     TSTB    @$TKS           ;;CHAR THERE?
13505  066006  100375                          BPL     7$              ;;IF NOT TRY AGAIN
13506
```

```
13507  066010  117746  113146                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
13508  066014  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13509
13510  066020  021627  000003                 CMP     (SP),#3         ;;IS IT A CONTROL-C?
13511  066024  001015                         BNE     9$              ;;BRANCH IF NOT
13512  066026  104401  066566                 TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
13513  066032  062706  000006                 ADD     #6,SP           ;;CLEAN UP STACK
13514  066036  123727  001151  000001         CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
13515  066044  001003                         BNE     8$              ;;BRANCH IF NO
13516  066046  012777  000100  113104         MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
13517  066054  000137  063404         8$:     JMP     SHUT2           ;;CONTROL-C RESTART
13518
13519
13520  066060  021627  000025         9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
13521  066064  001005                         BNE     10$             ;;BRANCH IF NOT
13522  066066  104401  066573                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
13523  066072  062706  000006         20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
13524  066076  000737                         BR      19$             ;;LET'S TRY IT AGAIN
13525
13526
13527  066100  021627  000015         10$:    CMP     (SP),#15        ;;IS IT A <CR>?
13528  066104  001022                         BNE     16$             ;;BRANCH IF NO
13529  066106  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
13530  066112  001403                         BEQ     11$             ;;BRANCH IF YES
13531  066114  016677  000002  113032         MOV     2(SP),@SWR      ;;SAVE NEW SWR
13532  066122  062706  000006         11$:    ADD     #6,SP           ;;CLEAR UP STACK
13533  066126  104401  001217         14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
13534  066132  123727  001151  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
13535  066140  001003                         BNE     15$             ;;BRANCH IF NOT
13536  066142  012777  000100  113010         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
13537  066150  000002         15$:    RTI                             ;;RETURN
13538  066152  004737  064526         16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
13539  066156  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
13540  066162  002420                         BLT     18$             ;;BRANCH IF YES
13541  066164  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
13542  066170  003015                         BGT     18$             ;;BRANCH IF YES
13543  066172  042726  000060                 BIC     #60,(SP)+       ;;STRIP-OFF ASCII
13544  066176  005766  000002                 TST     2(SP)           ;;IS THIS THE FIRST CHAR
13545  066202  001403                         BEQ     17$             ;;BRANCH IF YES
13546  066204  006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
13547  066206  006316                         ASL     (SP)            ;;   CHAR OVER TO MAKE
13548  066210  006316                         ASL     (SP)            ;;   ROOM FOR NEW ONE.
13549  066212  005266  000002         17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
13550  066216  056616  177776                 BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
13551  066222  000667                         BR      7$              ;;GET THE NEXT ONE
13552  066224  104401  001216         18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
13553  066230  000720                         BR      20$             ;;SIMULATE CONTROL-U
13554                                 .DSABL  LSB
13555
13556
13557                                 ;;*****************************************************************
13558                                 ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
13559                                 ;*CALL:
13560                                 ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
13561                                 ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
13562                                 ;*                              ;;WITH PARITY BIT STRIPPED OFF
```

```
13563                                   ;
13564
13565  066232  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
13566  066234  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE PS
13567  066242  005066  000004                   CLR     4(SP)           ;;GET READY FOR A CHARACTER
13568  066246  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
13569  066250  012746  066256                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
13570  066254  000002                           RTI                     ;;POP NEW PC AND PS
13571  066256                           64$:
13572  066256  005737  065372           1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
13573  066262  001775                           BEQ     1$
13574  066264  005337  065372                   DEC     $TKCNT          ;;DECREMENT THE COUNTER
13575  066270  117766  177102  000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
13576  066276  005237  065376                   INC     $TKQOUT         ;;UPDATE THE POINTER
13577  066302  023727  065376  065401           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
13578  066310  001003                           BNE     2$              ;;BRANCH IF NO
13579  066312  012737  065400  065376           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
13580  066320  000002           2$:             RTI                     ;;RETURN
13581                                   ;;*****************************************************************
13582                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
13583                                   ;*CALL:
13584                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
13585                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
13586                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
13587
13588  066322  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
13589  066324  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
13590  066326  012703  066556           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
13591  066332  022703  066566           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
13592  066336  101456                           BLOS    4$              ;;BR IF YES
13593  066340  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
13594  066342  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
13595  066344  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
13596  066350  001022                           BNE     5$              ;;BR IF NO
13597  066352  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
13598  066354  001007                           BNE     6$              ;;BR IF NO
13599  066356  112737  000134  066554           MOVB    #'\,9$          ;;TYPE A BACK SLASH
13600  066364  104401  066554                   TYPE    ,9$
13601  066370  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
13602  066374  005303                   6$:     DEC     R3              ;;BACKUP BY ONE
13603  066376  020327  066556                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
13604  066402  103434                           BLO     4$              ;;BR IF YES
13605  066404  111337  066554                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
13606  066410  104401  066554                   TYPE    ,9$             ;;GO TYPE
13607  066414  000746                           BR      2$              ;;GO READ ANOTHER CHAR.
13608  066416  005716                   5$:     TST     (SP)            ;;RUBOUT KEY SET?
13609  066420  001406                           BEQ     7$              ;;BR IF NO
13610  066422  112737  000134  066554           MOVB    #'\,9$          ;;TYPE A BACK SLASH
13611  066430  104401  066554                   TYPE    ,9$
13612  066434  005016                           CLR     (SP)            ;;CLEAR THE RUBOUT KEY
13613  066436  122713  000025           7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
13614  066442  001003                           BNE     8$              ;;BR IF NO
13615  066444  104401  066573                   TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U''
13616  066450  000726                           BR      1$              ;;GO START OVER
13617  066452  122713  000022           8$:     CMPB    #22,(R3)        ;;IS CHARACTER A ''^R''?
13618  066456  001011                           BNE     3$              ;;BRANCH IF NO
```

```
13619  066460  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
13620  066462  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
13621  066466  104401  066556                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
13622  066472  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
13623  066474  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
13624  066500  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
13625  066502  111337  066554          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
13626  066506  104401  066554                  TYPE    ,9$
13627  066512  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
13628  066516  001305                          BNE     2$              ;;LOOP IF NOT RETURN
13629  066520  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
13630  066524  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
13631  066530  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
13632  066532  012603                          MOV     (SP)+,R3        ;;RESTORE R3
13633  066534  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
13634  066536  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
13635  066544  012766  066556  000004          MOV     #$TTYIN,4(SP)
13636  066552  000002                          RTI                     ;;RETURN
13637  066554     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
13638  066555     000                          .BYTE   0               ;;TERMINATOR
13639  066556  000010              $TTYIN:     .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
13640  066566  041536  005015     000  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL "C"
13641  066573     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
13642  066600  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
13643  066605     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
13644  066612  036440  000040
13645  066616  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
13646  066624  020075     000
13647          066630                  .EVEN
13648                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
13649
13650                                  ;;**********************************************************************
13651                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
13652                                  ;*CHANGE IT TO BINARY.
13653                                  ;*CALL:
13654                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
13655                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
13656                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
13657
13658  066630  011646              $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
13659  066632  016666  000004  000002      MOV     4(SP),2(SP)     ;;INPUT NUMBER
13660  066640  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13661  066642  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13662  066644  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13663  066646  104412              1$:     RDLIN                   ;;READ AN ASCIZ LINE
13664  066650  012600                      MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
13665  066652  005001                      CLR     R1              ;;CLEAR DATA WORD
13666  066654  005002                      CLR     R2
13667  066656  112046              2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
13668  066660  001412                      BEQ     3$              ;;IF ZERO GET OUT
13669  066662  006301                      ASL     R1              ;;*2
13670  066664  006102                      ROL     R2
13671  066666  006301                      ASL     R1              ;;*4
13672  066670  006102                      ROL     R2
13673  066672  006301                      ASL     R1              ;;*8
13674  066674  006102                      ROL     R2
```

```
                        READ AN OCTAL NUMBER FROM THE TTY

13675  066676  042716  177770            BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
13676  066702  062601                    ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
13677  066704  000764                    BR      2$              ;;LOOP
13678  066706  005726            3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
13679  066710  010166  000012            MOV     R1,12(SP)       ;;SAVE THE RESULT
13680  066714  010237  066730            MOV     R2,$HIOCT
13681  066720  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
13682  066722  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
13683  066724  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
13684  066726  000002                    RTI                     ;;RETURN
13685  066730  000000            $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
13686                             .SBTTL  TRAP DECODER
13687
13688                             ;;**************************************************************
13689                             ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
13690                             ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
13691                             ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
13692                             ;*GO TO THAT ROUTINE.
13693
13694  066732  016646  000002    $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
13695  066736  042716  000020            BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
13696  066742  012746  066750            MOV     #1$,-(SP)       ;; T-BIT TRAPS
13697  066746  000002                    RTI                     ;;SET THE NEW STATUS
13698  066750  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
13699  066752  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
13700  066756  005740                    TST     -(R0)           ;;BACKUP BY 2
13701  066760  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
13702  066762  006300                    ASL     R0              ;;POSITION FOR INDEXING
13703  066764  016000  067004            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
13704  066770  000200                    RTS     R0              ;;GO TO ROUTINE
13705
13706
13707                             ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
13708
13709  066772  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
13710  066774  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
13711  067002  000002                    RTI                     ;;RESTORE THE PSW
13712
13713                             .SBTTL  TRAP TABLE
13714
13715                             ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
13716                             ;*BY THE "TRAP" INSTRUCTION.
13717
13718                             ;       ROUTINE
13719                             ;       -------
13720  067004  066772            $TRPAD: .WORD   $TRAP2
13721  067006  064314                    $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
13722  067010  064112                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
13723  067012  064066                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
13724  067014  064126                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
13725  067016  063642                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
13726  067020  063566                    $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
13727
13728  067022  065760                    $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
13729
13730  067024  065670                    $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
```

```
13731  067026  066232                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
13732  067030  066322                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
13733  067032  066630                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
13734  067034  063472                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
13735  067036  063530                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
13736                                  .SBTTL  POWER DOWN AND UP ROUTINES
13737
13738                                  ;;**********************************************************************
13739                                  ;POWER DOWN ROUTINE
13740  067040  012737  067200  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
13741  067046  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
13742  067054  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13743  067056  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13744  067060  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13745  067062  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
13746  067064  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
13747  067066  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
13748  067070  017746  112060                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
13749  067074  010637  067204                  MOV     SP,$SAVR6       ;;SAVE SP
13750  067100  012737  067112  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
13751  067106  000000                          HALT
13752  067110  000776                          BR      .-2             ;;HANG UP
13753
13754                                  ;;**********************************************************************
13755                                  ;POWER UP ROUTINE
13756  067112  012737  067200  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
13757  067120  013706  067204                  MOV     $SAVR6,SP       ;;GET SP
13758  067124  005037  067204                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
13759  067130  005237  067204          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
13760  067134  001375                          BNE     1$              ;;OF  WORD
13761  067136  012677  112012                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
13762  067142  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
13763  067144  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
13764  067146  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
13765  067150  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
13766  067152  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
13767  067154  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
13768  067156  012737  067040  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
13769  067164  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
13770  067172  104401                          TYPE                    ;REPORT THE POWER FAILURE
13771  067174  067206                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
13772  067176  000002                          RTI
13773  067200  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
13774  067202  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
13775  067204  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
13776  067206  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
13777  067214  000122
13778                                          .EVEN
13779                                  .SBTTL  APT COMMUNICATIONS ROUTINE
13780
13781                                  ;;**********************************************************************
13782  067216  112737  000001  067462  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
13783  067224  112737  000001  067460  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
13784  067232  000403                          BR      $ATYC
13785  067234  112737  000001  067462  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
13786  067242                          $ATYC:
```

```
13787  067242  010046                        MOV     R0,-(SP)          ;;PUSH R0 ON STACK
13788  067244  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
13789  067246  105737  067460                TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
13790  067252  001450                        BEQ     5$                ;;IF NOT:  BR
13791  067254  122737  000001  001242        CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
13792  067262  001031                        BNE     3$                ;;IF NOT:  BR
13793  067264  132737  000100  001243        BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
13794  067272  001425                        BEQ     3$                ;;IF NOT:  BR
13795  067274  017600  000004                MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
13796  067300  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
13797  067306  005737  001222        1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
13798  067312  001375                        BNE     1$                ;;IF NOT:  WAIT
13799  067314  010037  001236                MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
13800  067320  105720                2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
13801  067322  001376                        BNE     2$
13802  067324  163700  001236                SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
13803  067330  006200                        ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
13804  067332  010037  001240                MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
13805  067336  012737  000004  001222        MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
13806  067344  000413                        BR      5$
13807  067346  017637  000004  067372  3$:   MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
13808  067354  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
13809  067362  013746  177776                MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
13810  067366  004737  064314                JSR     PC,$TYPE          ;;CALL TYPE MACRO
13811  067372  000000                4$:     .WORD   0
13812  067374                        5$:
13813  067374  105737  067462        10$:    TSTB    $FFLG             ;;SHOULD REPORT FATAL ERROR?
13814  067400  001416                        BEQ     12$               ;;IF NOT:  BR
13815  067402  005737  001242                TST     $ENV              ;;RUNNING UNDER APT?
13816  067406  001413                        BEQ     12$               ;;IF NOT:  BR
13817  067410  005737  001222        11$:    TST     $MSGTYPE          ;;FINISHED LAST MESSAGE?
13818  067414  001375                        BNE     11$               ;;IF NOT:  WAIT
13819  067416  017637  000004  001224        MOV     @4(SP),$FATAL     ;;GET ERROR #
13820  067424  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
13821  067432  005237  001222                INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR
13822  067436  105037  067462        12$:    CLRB    $FFLG             ;;CLEAR FATAL FLAG
13823  067442  105037  067461                CLRB    $LFLG             ;;CLEAR LOG FLAG
13824  067446  105037  067460                CLRB    $MFLG             ;;CLEAR MESSAGE FLAG
13825  067452  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
13826  067454  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
13827  067456  000207                        RTS     PC                ;;RETURN
13828  067460     000          $MFLG:  .BYTE   0                 ;;MESSG. FLAG
13829  067461     000          $LFLG:  .BYTE   0                 ;;LOG FLAG
13830  067462     000          $FFLG:  .BYTE   0                 ;;FATAL FLAG
13831          067464                  .EVEN
13832          000200          APTSIZE=200
13833          000001          APTENV=001
13834          000100          APTSPOOL=100
13835          000040          APTCSUP=040
13836
13837                          .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

067464                          SCTMSG:
067464  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
067546  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
067574     051                  CLSPRN: .ASCII  @)@
067575     075     000          EQUALS: .ASCIZ  @=@
067577     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
067603     077     000          QSTMRK: .ASCIZ  @?@
067605                          HELPQST:
067605     015  006412  052012  .ASCII  <CR><LF><CR><LF>/THIS PROGRAM SHOULD BE HALTED BY TYPE ^C./
067662  005015  005015  044124  .ASCII  <CR><LF><CR><LF>/THE PROGRAM WILL BE HALTED AT END OF PASS/
067737     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
067773                          UBUSQST:
067773     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@
070072  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
070131     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
070156  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
070205     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
070237     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
070267     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
070313     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
070343     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
070377     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
070424                          CNSL07:
070424  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
070502  047040  046525  042502  .ASCII  @ NUMBER(S)@
070514  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
070563     015  041412  040510  XDPMG:  .ASCII  <CR><LF>/CHANGE XXDP PACK,CLEAR LOC 40/
070622  005015  042522  052123          .ASCIZ  <CR><LF>/RESTART THE PROGRAM/
070650  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.

070674                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
070674  020000          .WORD   OPI                             ;NOP
070676  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (2)
070700  132000          .WORD   ATA!OPI!IVC!IAE                 ;SEEK
070702  130000          .WORD   ATA!OPI!IVC                     ;RECALIBRAIE
070704  020000          .WORD   OPI                             ;DRIVE CLEAR
070706  030000          .WORD   OPI!IVC                         ;RELEASE
070710  130000          .WORD   OPI!ATA!IVC                     ;OFFSET
070712  130000          .WORD   OPI!ATA!IVC                     ;RETURN TO CENTERLINE
070714  020000          .WORD   OPI                             ;READ IN PRESET
070716  020000          .WORD   OPI                             ;PACK ACKNOWLEDGE
070720  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (24)
070722  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (26)
070724  132000          .WORD   ATA!OPI!IVC!IAE                 ;SEARCH
070726  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (32)
070730  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (34)
070732  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (36)
070734  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (40)
070736  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (42)
070740  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (44)
070742  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (46)
070744  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK DATA
070746  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK HEADER AND DATA
070750  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (54)
070752  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (56)
070754  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE         ;WRITE DATA
070756  037000          .WORD   OPI!IVC!WLE!IAE!AOE             ;WRITE HEADER AND DATA
070760  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (64)
070762  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (66)
070764  033300          .WORD   OPI!JVC!IAE!AOE!HCE!ECH         ;READ DATA
070766  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ HEADER AND DATA
070770  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (74)
070772  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (76)
```

```
                                .SBTTL  ATTENTION (ATA) TABLE

070774  001             ATNTBL: .BYTE   1.
070775  002                     .BYTE   2.
070776  004                     .BYTE   4.
070777  010                     .BYTE   8.
071000  020                     .BYTE   16.
071001  040                     .BYTE   32.
071002  100                     .BYTE   64.
071003  200                     .BYTE   128.
```

```
                                .SBTTL  DATA PATTERN TABLE

071004                          RGDTPT:
071004                          MIXED:
071004  000000                          .WORD   0.
071006  000001                          .WORD   1.
071010  000003                          .WORD   3.
071012  000007                          .WORD   7.
071014  000017                          .WORD   15.
071016  000037                          .WORD   31.
071020  000077                          .WORD   63.
071022  000177                          .WORD   127.
071024  000377                          .WORD   255.
071026  000777                          .WORD   511.
071030  001777                          .WORD   1023.
071032  003777                          .WORD   2047.
071034  007777                          .WORD   4095.
071036  017777                          .WORD   8191.
071040  037777                          .WORD   16383.
071042  077777                          .WORD   32767.
071044  177777                  ONES:   .WORD   65535.
071046  177777                          .WORD   65535.
071050  077777                          .WORD   32767.
071052  037777                          .WORD   16383.
071054  017777                          .WORD   8191.
071056  007777                          .WORD   4095.
071060  003777                          .WORD   2047.
071062  001777                          .WORD   1023.
071064  000777                          .WORD   511.
071066  000377                          .WORD   255.
071070  000177                          .WORD   127.
071072  000077                          .WORD   63.
071074  000037                          .WORD   31.
071076  000017                          .WORD   15.
071100  000007                          .WORD   7.
071102  000003                          .WORD   3.
071104  000001                          .WORD   1.
071106  000000                  ZEROS:  .WORD   0.
071110  000000                          .WORD   0.
071112  000001                          .WORD   1.
071114  000002                          .WORD   2.
071116  000004                          .WORD   4.
071120  000010                          .WORD   8.
071122  000020                          .WORD   16.
071124  000040                          .WORD   32.
071126  000100                          .WORD   64.
071130  000200                          .WORD   128.
071132  000400                          .WORD   256.
071134  001000                          .WORD   512.
071136  002000                          .WORD   1024.
071140  004000                          .WORD   2048.
071142  010000                          .WORD   4096.
071144  020000                          .WORD   8192.
071146  040000                          .WORD   16384.
071150  100000                          .WORD   32768.
071152  100000                          .WORD   32768.
```

```
071154  040000                          .WORD   16384.
071156  020000                          .WORD   8192.
071160  010000                          .WORD   4096.
071162  004000                          .WORD   2048.
071164  002000                          .WORD   1024.
071166  001000                          .WORD   512.
071170  000400                          .WORD   256.
071172  000200                          .WORD   128.
071174  000100                          .WORD   64.
071176  000040                          .WORD   32.
071200  000020                          .WORD   16.
071202  000010                          .WORD   8.
071204  000004                          .WORD   4.
071206  000002                          .WORD   2.
071210  000001                          .WORD   1.
071212  000000                          .WORD   0.
071214  177777                          .WORD   65535.
071216  177776                          .WORD   65534.
071220  177774                          .WORD   65532.
071222  177770                          .WORD   65528.
071224  177760                          .WORD   65520.
071226  177740                          .WORD   65504.
071230  177700                          .WORD   65472.
071232  177600                          .WORD   65408.
071234  177400                          .WORD   65280.
071236  177000                          .WORD   65024.
071240  176000                          .WORD   64512.
071242  174000                          .WORD   63488.
071244  170000                          .WORD   61440.
071246  160000                          .WORD   57344.
071250  140000                          .WORD   49152.
071252  100000                          .WORD   32768.
071254  000000                          .WORD   0.
071256  000000                          .WORD   0.
071260  100000                          .WORD   32768.
071262  140000                          .WORD   49152.
071264  160000                          .WORD   57344.
071266  170000                          .WORD   61440.
071270  174000                          .WORD   63488.
071272  176000                          .WORD   64512.
071274  177000                          .WORD   65024.
071276  177400                          .WORD   65280.
071300  177600                          .WORD   65408.
071302  177700                          .WORD   65472.
071304  177740                          .WORD   65504.
071306  177760                          .WORD   65520.
071310  177770                          .WORD   65528.
071312  177774                          .WORD   65532.
071314  177776                          .WORD   65534.
071316  177777                          .WORD   65535.
071320  125252          -       EARLY:  .WORD   43690.
071322  152525                          .WORD   43690./2
071324  125252                          .WORD   43690.
071326  177777                          .WORD   65535.
071330  177776                          .WORD   65534.
071332  177775                          .WORD   65533.
```

```
071334  177773                  .WORD   65531.
071336  177767                  .WORD   65527.
071340  177757                  .WORD   65519.
071342  177737                  .WORD   65503.
071344  177677                  .WORD   65471.
071346  177577                  .WORD   65407.
071350  177377                  .WORD   65279.
071352  176777                  .WORD   65023.
071354  175777                  .WORD   64511.
071356  173777                  .WORD   63487.
071360  167777                  .WORD   61439.
071362  157777                  .WORD   57343.
071364  137777                  .WORD   49151.
071366  077777                  .WORD   32767.
071370  077777                  .WORD   32767.
071372  137777                  .WORD   49151.
071374  157777                  .WORD   57343.
071376  167777                  .WORD   61439.
071400  173777                  .WORD   63487.
071402  175777                  .WORD   64511.
071404  176777                  .WORD   65023.
071406  177377                  .WORD   65279.
071410  177577                  .WORD   65407.
071412  177677                  .WORD   65471.
071414  177737                  .WORD   65503.
071416  177757                  .WORD   65519.
071420  177767                  .WORD   65527.
071422  177773                  .WORD   65531.
071424  177775                  .WORD   65533.
071426  177776                  .WORD   65534.
071430  177777                  .WORD   65535.
071432                  ENRGDT:
```

```
                                .SBTTL  ERROR MESSAGE TABLE

071432  076026  000000          EMT1:   .WORD   EMS1,0
071436  076075  076112  000000  EMT2:   .WORD   EMS2,EMS3,0
071444  076075  076155  000000  EMT3:   .WORD   EMS2,EMS4,0
071452  076220  076250  000000  EMT4:   .WORD   EMS5,EMS6,0
071460  076220  076362  000000  EMT5:   .WORD   EMS5,EMS10,0
071466  103055  100243  000000  EMT6:   .WORD   EMS167,EMS64,0
071474  101011  103102  000000  EMT7:   .WORD   EMS110,EMS170,0
071502  076315  000000          EMT10:  .WORD   EMS7,0
071506  076362  000000          EMT11:  .WORD   EMS10,0
071512  076424  076435  000000  EMT12:  .WORD   EMS11,EMS12,0
071520  076476  076507  076520  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
071532  076572  100243  000000  EMT14:  .WORD   EMS17,EMS64,0
071540  076424  076655  000000  EMT15:  .WORD   EMS11,EMS21,0
071546  076424  076700  077027  EMT16:  .WORD   EMS11,EMS22,EMS27,0
071556  076424  076714  077040  EMT17:  .WORD   EMS11,EMS23,EMS30,0
071566  076424  076742  077040  EMT20:  .WORD   EMS11,EMS24,EMS30,0
071576  076424  076771  077027  EMT21:  .WORD   EMS11,EMS25,EMS27,0
071606  076424  077006  077027  EMT22:  .WORD   EMS11,EMS26,EMS27,0
071616  076424  077050  077040  EMT23:  .WORD   EMS11,EMS31,EMS30,0
071626  076424  077077  077040  EMT24:  .WORD   EMS11,EMS32,EMS30,0
071636  076424  077126  077040  EMT25:  .WORD   EMS11,EMS33,EMS30,0
071646  076424  077154  077040  EMT26:  .WORD   EMS11,EMS34,EMS30,0
071656  076424  077225  077040  EMT27:  .WORD   EMS11,EMS35,EMS30,0
071666  076424  077254  077040  EMT30:  .WORD   EMS11,EMS36,EMS30,0
071676  076424  077303  077040  EMT31:  .WORD   EMS11,EMS37,EMS30,0
071706  076424  077331  077040  EMT32:  .WORD   EMS11,EMS40,EMS30,0
071716  076424  077360  077040  EMT33:  .WORD   EMS11,EMS41,EMS30,0
071726  076424  077406  077040  EMT34:  .WORD   EMS11,EMS42,EMS30,0
071736  076424  077435  077040  EMT35:  .WORD   EMS11,EMS43,EMS30,0
071746  076424  077464  077040  EMT36:  .WORD   EMS11,EMS44,EMS30,0
071756  076424  077537  077040  EMT37:  .WORD   EMS11,EMS45,EMS30,0
071766  100327  076632  000000  EMT40:  .WORD   EMS66,EMS20,0
071774  100515  102223  100523  EMT41:  .WORD   EMS75,EMS141,EMS76,0
072004  102314  102324  100451  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
072016  077631  077747  100523  EMT43:  .WORD   EMS47,EMS53,EMS76,0
072026  100554  077747  100523  EMT44:  .WORD   EMS77,EMS53,EMS76,0
072036  100602  077747  100523  EMT45:  .WORD   EMS100,EMS53,EMS76,0
072046  100630  077747  100523  EMT46:  .WORD   EMS101,EMS53,EMS76,0
072056  100261  100243  000000  EMT47:  .WORD   EMS65,EMS64,0
072064  076476  076520  100215  EMT50:  .WORD   EMS13,EMS15,EMS63,0
072074  076424  077602  077040  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
072106  077631  077747  100377  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
072124  077631  077747  100377  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
072142  077660  077747  100377  EMT54:  .WORD   EMS50,EMS53,EMS67,0
072152  102251  102266  077747  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
072164  077707  100451  100377  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
072202  103055  100461  100377  EMT57:  .WORD   EMS167,EMS73,EMS67,0
072212  100032  100377  101211  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
072230  100436  100032  100377  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
072250  102202  100377  101211  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
072264  000000                  EMT63:  .WORD
072266  102407  102452  100424  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
072306  077735  103502  103663  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
072326  103014  100705  100451  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
072340  103014  100705  100451  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
072352  077602  076632  102707  EMT70:  .WORD   EMS46,EMS20,EMS163,0
072362  100436  100630  102707  EMT71:  .WORD   EMS71,EMS101,EMS163,0
072372  077631  100451  102707  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
072410  077631  100451  102707  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
072426  100032  077747  102707  EMT74:  .WORD   EMS56,EMS53,EMS163,0
072436  100436  100032  102707  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
072456  077660  077747  102707  EMT76:  .WORD   EMS50,EMS53,EMS163,0
072466  102251  102266  077747  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
072500  100515  102223  102707  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
072516  100515  102407  102707  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
072534  1^3055  100461  102707  EMT102: .WORD   EMS167,EMS73,EMS163,0
072544  102372  102426  100476  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
072556  077707  100476  102707  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
072574  103014  100602  102707  EMT105: .WORD   EMS165,EMS100,EMS163,0
072604  103014  100554  102707  EMT106: .WORD   EMS165,EMS77,EMS163,0
072614  102314  102324  077747  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
072634  101011  101051  101216  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
072646  101135  076155  000000  EMT111: .WORD   EMS113,EMS4,0
072654  101135  076112  000000  EMT112: .WORD   EMS113,EMS3,0
072662  101011  101211  101216  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
072676  101135  101270  000000  EMT114: .WORD   EMS113,EMS120,0
072704  101305  101745  101771  EMT115: .WORD   EMS121,EMS132,EMS133,0
072714  101350  101745  101771  EMT116: .WORD   EMS122,EMS132,EMS133,0
072724  101405  101745  101771  EMT117: .WORD   EMS123,EMS132,EMS133,0
072734  101450  101745  101771  EMT120: .WORD   EMS124,EMS132,EMS133,
072744  101502  101745  101771  EMT121: .WORD   EMS125,EMS132,EMS133,0
072754  101545  101745  101771  EMT122: .WORD   EMS126,EMS132,EMS133,0
072764  102145  101745  101771  EMT123: .WORD   EMS137,EMS132,EMS133,0
072774  101647  101745  101771  EMT124: .WORD   EMS130,EMS132,EMS133,0
073004  101705  101745  101771  EMT125: .WORD   EMS131,EMS132,EMS133,0
073014  101305  101745  102014  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
073026  101350  101745  102014  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
073040  101405  101745  102014  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
073052  101450  101745  102014  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
073064  101502  101745  102014  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
073076  101545  101745  102014  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
073110  102145  101745  102014  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
073122  101647  101745  102014  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
073134  101705  101745  102014  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
073146  101305  101745  102056  EMT137: .WORD   EMS121,EMS132,EMS135,0
073156  101405  101745  102056  EMT140: .WORD   EMS123,EMS132,EMS135,0
073166  101305  101745  102120  EMT141: .WORD   EMS121,EMS132,EMS136,0
073176  102145  101745  102120  EMT142: .WORD   EMS137,EMS132,EMS136,0
073206  101450  101745  102120  EMT143: .WORD   EMS124,EMS132,EMS136,0
073216  101502  101745  102120  EMT144: .WORD   EMS125,EMS132,EMS136,0
073226  101545  101745  102120  EMT145: .WORD   EMS126,EMS132,EMS136,0
073236  101705  101745  102120  EMT146: .WORD   EMS131,EMS132,EMS136,0
073246  102654  101745  102120  EMT147: .WORD   EMS162,EMS132,EMS136,0
073256  101647  101745  102120  EMT150: .WORD   EMS130,EMS132,EMS136,0
073266  102202  101211  102223  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
073300  102251  101211  102266  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
073312  102314  102324  102353  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
073330  102314  102324  076632  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
073346  102^07  102452  102520  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
073360  10237?  102426  102520  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
073372  102372  102426  102554  EMT157: .WORD   EMS147,EMS151,EMS150,EMS157,0
073404  102624  102554  102607  EMT160: .WORD   EMS161,EMS156,EMS160,0
073414  076771  077027  102554  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
073426  077006  077027  102554  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
073440  077631  102707  102202  EMT163: .WORD   EMS47,EMS163,EMS140,0
073450  077631  076632  000000  EMT164: .WORD   EMS47,EMS20,0
073456  100032  076632  000000  EMT165: .WORD   EMS56,EMS20,0
073464  077602  076632  000000  EMT166: .WORD   EMS46,EMS20,0
073472  104203  076632  000000  EMT167: .WORD   EMS224,EMS20,0
073500  103055  102520  103030  EMT170: .WORD   EMS167,EMS154,EMS166,0
073510  103055  102520  102572  EMT171: .WORD   EMS167,EMS154,EMS157,0
073520  103055  102520  102534  EMT172: .WORD   EMS167,EMS154,EMS155,0
073530  103131  101745  101771  EMT173: .WORD   EMS171,EMS132,EMS133,0
073540  103131  101745  102014  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
073552  101135  103304  000000  EMT175: .WORD   EMS113,EMS177,0
073560  103326  103343  000000  EMT176: .WORD   EMS200,EMS201,0
073566  103441  102223  077040  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
073600  103441  077707  077040  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
073612  103441  103454  077040  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
073624  103441  077660  077040  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
073636  103441  102266  077040  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
073650  102407  100476  103411  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
073666  077660  100705  100451  EMT205: .WORD   EMS50,EMS103,EMS72,0
073676  100714  100461  103411  EMT206: .WORD   EMS104,EMS73,EMS202,0
073706  100515  102223  101211  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
073720  100515  102407  000000  EMT210: .WORD   EMS75,EMS150,0
073726  077707  100451  101211  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
073742  102251  077747  101211  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
073756  077660  100705  077747  EMT213: .WORD   EMS50,EMS103,EMS53,0
073766  077735  103502  103030  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
074006  077735  101246  100032  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
074020  100032  077040  100243  EMT216: .WORD   EMS56,EMS30,EMS64,0
074030  077602  077040  100243  EMT217: .WORD   EMS46,EMS30,EMS64,0
074040  077050  077040  100243  EMT220: .WORD   EMS31,EMS30,EMS64,0
074050  077631  077040  100243  EMT221: .WORD   EMS47,EMS30,EMS64,0
074060  077735  101246  100554  EMT222: .WORD   EMS52,EMS117,EMS77,0
074070  077735  101246  100215  EMT223: .WORD   EMS52,EMS117,EMS63,0
074100  000000                  EMT224: .WORD
074102  000000                  EMT225: .WORD
074104  000000                  EMT226: .WORD
074106  000000                  EMT227: .WORD
074110  000000                  EMT230: .WORD
074112  000000                  EMT231: .WORD
074114  000000                  EMT232: .WORD
074116  000000                  EMT233: .WORD
074120  000000                  EMT234: .WORD
074122  000000                  EMT235: .WORD
074124  000000                  EMT236: .WORD
074126  000000                  EMT237: .WORD
074130  000000                  EMT240: .WORD
074132  000000                  EMT241: .WORD
074134  000000                  EMT242: .WORD
074136  000000                  EMT243: .WORD
074140  000000                  EMT244: .WORD
074142  000000                  EMT245: .WORD
074144  103055  101745  103541  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
074154  103055  101745  103566  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
074166  103055  103577  103566  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
074202  103615  103640  000000  EMT251: .WORD   EMS212,EMS213,0
074210  103014  103615  000000  EMT252: .WORD   EMS165,EMS212,0
074216  102372  102426  102554  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
074234  103441  100630  077040  EMT254: .WORD   EMS203,EMS101,EMS30,0
074244  103441  103055  077040  EMT255: .WORD   EMS203,EMS167,EMS30,0
074254  103441  100602  077040  EMT256: .WORD   EMS203,EMS100,EMS30,0
074264  076424  077602  077040  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
074276  077735  101246  100032  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
074310  077735  103502  104005  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
074330  100714  100461  103411  EMT262: .WORD   EMS104,EMS73,EMS202,0
074340  077660  077747  100656  EMT263: .WORD   EMS50,EMS53,EMS102,0
074350  100032  077747  100656  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
074370  100436  100032  100656  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
074410  102251  102266  077747  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
074422  077660  102353  101211  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
074440  077631  077747  100656  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
074454  077631  077747  100656  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
074472  100515  102223  100656  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
074510  077707  100476  100656  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
074526  100215  077747  100107  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
074544  101216  077747  077360  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
074556  077631  077747  100107  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
074572  077631  077747  100107  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
074610  100032  077747  100107  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
074630  100436  100032  077747  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
074652  103014  100602  100705  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
074664  103014  100630  100705  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
074676  103014  100554  100705  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
074710  077602  077040  100243  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
074722  077660  077747  100107  EMT306: .WORD   EMS50,EMS53,EMS57,0
074732  077660  102353  101211  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
074752  102251  102266  077747  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
074764  100714  100705  077747  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
074776  100743  100705  077747  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
075010  102314  102324  100705  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
075030  077406  100705  077747  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
075042  077050  100705  077747  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
075054  100436  077050  100705  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
075066  077435  100705  100107  EMT317: .WORD   EMS43,EMS103,EMS57,0
075076  077537  100705  100107  EMT320: .WORD   EMS45,EMS103,EMS57,0
075106  077464  100705  100107  EMT321: .WORD   EMS44,EMS103,EMS57,0
075116  100772  076632  000000  EMT322: .WORD   EMS106,EMS20,0
075124  077254  100705  100107  EMT323: .WORD   EMS36,EMS103,EMS57,0
075134  103204  077254  100705  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
075146  103164  077254  100705  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
075160  076476  103221  076520  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
075176  102372  102426  100476  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
075210  100327  077747  103227  EMT330: .WORD   EMS66,EMS53,EMS175,0
075220  077126  100705  077747  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
075232  077331  100705  077747  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
075244  077707  100476  100107  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
075262  100515  102223  100107  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
075300  100515  102407  102452  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
075320  100135  100150  100177  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
075330  101216  101246  077154  EMT337: .WORD   EMS116,EMS117,EMS34,0
075340  077154  077747  077761  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
075352  100003  077154  000000  EMT341: .WORD   EMS55,EMS34,0
075360  077735  101246  100032  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
075372  077735  101246  077537  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
075404  077735  101246  077464  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
075416  077735  101246  104025  EMT345: .WORD   EMS52,EMS117,EMS221,0
075426  100772  076632  101211  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
075442  077735  103502  104075  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
075454  100515  102407  100656  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
075472  103055  100461  100656  EMT351: .WORD   EMS167,EMS73,EMS102,0
075502  103701  000000          EMT352: .WORD   EMS215,0
075506  103752  103441  103722  EMT353: .WORD   EMS217,EMS203,EMS216,0
075516  104025  076632  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
075524  104255  105061  105136  EMT1:    .WORD   EM1,STSH1,STSH2,STSH4,0
075536  105061  105136  105263  EMT2:    .WORD   STSH1,STSH2,STSH4,0

075546  104274  000000          EMT110:  .WORD   EM110,0
075552  104303  000000          EMT111:  .WORD   EM111,0

075556  104322  000000          EMT114:  .WORD   EM114,0
075562  104351  105061  105136  EMT223:  .WORD   EM223,STSH1,STSH2,STSH4,0
075574  104377  105061  105136  EMT256:  .WORD   EM256,STSH1,STSH2,STSH4,0

075606  104454  105061  105136  EMT336:  .WORD   EM336,STSH1,STSH2,STSH4,0
075620  104513  105061  105136  EMT337:  .WORD   EM337,STSH1,STSH2,STSH4,0
075632  104652  105061  105136  EMT344:  .WORD   EM344,STSH1,STSH2,STSH4,0

075644  105012  000000          EMT353:  .WORD   EM353,0
```

```
075650  105322  105416  105434  EDT1:   .WORD   ED1,STSD1,STSD2,STSD4
075660  105416  105434  105466  EDT2:   .WORD   STSD1,STSD2,STSD4

075666  105330                  EDT110: .WORD   ED110
075670  105334                  EDT111: .WORD   ED111

075672  105342                  EDT114: .WORD   ED114
075674  105352  105416  105434  EDT223: .WORD   ED223,STSD1,STSD2,STSD4

075704  105362  105416  105434  EDT336: .WORD   ED336,STSD1,STSD2,STSD4
075714  105374  105416  105434  EDT337: .WORD   ED337,STSD1,STSD2,STSD4
075724  105374  105416  105434  EDT344: .WORD   ED337,STSD1,STSD2,STSD4,0

075736  105406                  EDT353: .WORD   ED353
```

```
075740  105501  105517  105517  EFT1:   .WORD   EF111,STSF,STSF,STSF
075750  105517  105517  105517  EFT2:   .WORD   STSF,STSF,STSF

075756  105500                  EFT110: .WORD   EF110
075760  105501                  EFT111: .WORD   EF111

075762  105503                  EFT114: .WORD   EF114
075764  105503  105517  105517  EFT223: .WORD   EF114,STSF,STSF,STSF

075774  105506  105517  105517  EFT336: .WORD   EF336,STSF,STSF,STSF
076004  105506  105517  105517  EFT337: .WORD   EF336,STSF,STSF,STSF
076014  105506  105517  105517  EFT344: .WORD   EF336,STSF,STSF,STSF

076024  105503                  EFT353: .WORD   EF114
```

```
                            .SBTTL  ERROR MESSAGE STRINGS

076026 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
076075    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
076112 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
076155    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
076220 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED, @
076250 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
076315    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
076362 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
076424 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
076435    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
076476 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
076507    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
076520 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
076542 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
076572 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
076632 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
076655    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
076700 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
076714 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
076742 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
076771    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
077006 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
077027    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
077040 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
077050 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
077077    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
077126 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
077154 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
077225    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
077254 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
077303    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
077331    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
077360 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
077406 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
077435    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
077464 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
077537    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
077602 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
077631    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
077660 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
077707    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
077735    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
077747    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
077761    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
100003    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
100032 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
100107    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
100135    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
100150 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
100177    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
100215    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
100243    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
100261    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
100327    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
100377     104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
100424  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
100436  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
100451     111  020123  042523  EMS72:  .ASCIZ  @IS SET @
100461     104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
100476  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
100515     114  051517  020124  EMS75:  .ASCIZ  @LOST @
100523     104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
100554  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
100602  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
100630  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
100656  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
100705     105  051122  051117  EMS103: .ASCIZ  @ERROR @
100714  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
100743     042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
100772  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
101011     102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
101040  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
101051     127  042510  020116  EMS112: .ASCII  @WHEN READING/WRITING RH REGISTERS @
101113     101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
101135     124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
101165     116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
101211     040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
101216  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
101246  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
101270  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
101305     103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
101350  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
101405     103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
101450  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
101502  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
101545     115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
101610  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
101647     105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
101705     115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
101745     116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
101771     125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
102014  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
102056  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
102120  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
102145     104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
102202  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
102223     042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
102251     104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
102266  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
102314  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
102324  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
102353     123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
102372  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
102407     040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
102426  047442  021115  024040  EMS151: .ASCIZ  @"DM" (RMDS, BIT 0) @
102452  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
102476  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
102520  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
102534  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
102554  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
102572 052122 020103 047503 EMS157: .ASCIZ @RTC COMMAND @
102607    122 050111 041440 EMS160: .ASCIZ @RIP COMMAND @
102624 043117 051506 052105 EMS161: .ASCIZ @OFFSET REGISTER (RMOF) @
102654 051105 047522 020122 EMS162: .ASCIZ @ERROR REGISTER #2, RMER2, @
102707    104 051125 047111 EMS163: .ASCIZ @DURING RECALIBRATE @
102733    111 020123 047111 EMS164: .ASCII @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
103002 054503 044514 042116       .ASCIZ @CYLINDER @
103014 047125 054105 042520 EMS165: .ASCIZ @UNEXPECTED @
103030 042522 040503 044514 EMS166: .ASCIZ @RECALIBRATE COMMAND @
103055    042 052101 021101 EMS167: .ASCIZ @"ATA" (RMDS, BIT15) @
103102 044127 047105 051040 EMS170: .ASCIZ @WHEN READING REGISTER @
103131    105 051122 051117 EMS171: .ASCIZ @ERROR REGISTER #2, RMER2, @
103164 047516 051116 041505 EMS172: .ASCIZ @NONRECOVERABLE @
103204 042522 047503 042526 EMS173: .ASCIZ @RECOVERABLE @
103221    104 052101 020101 EMS174: .ASCIZ @DATA @
103227    104 051125 047111 EMS175: .ASCIZ @DURING WRITE COMMAND @
103255    042 050117 021105 EMS176: .ASCIZ @"OPE" (RMER2, BIT 13) @
103304 047111 053440 044522 EMS177: .ASCIZ @IN WRITE PROTECT @
103326 040503 020116 047516 EMS200: .ASCIZ @CAN NOT SET @
103343    104 040511 047107 EMS201: .ASCIZ @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
103411    104 051125 047111 EMS202: .ASCIZ @DURING DIAGNOSTIC MODE @
103441    111 041516 051117 EMS203: .ASCIZ @INCORRECT @
103454 053442 046122 020042 EMS204: .ASCIZ @"WRL" (RMDS, BIT 11) @
103502 054105 041505 052125 EMS205: .ASCIZ @EXECUTED @
103514 044527 044124 041440 EMS206: .ASCIZ @WITH COMP ERROR SET @
103541    042 047507 020042 EMS207: .ASCIZ @"GO" (RMCS1, BIT 0) @
103566 051127 052111 047111 EMS210: .ASCIZ @WRITING @
103577    127 051501 051040 EMS211: .ASCIZ @WAS RESET BY @
103615    120 047522 051107 EMS212: .ASCIZ @PROGRAM INTERRUPT @
103640 040527 020123 047516 EMS213: .ASCIZ @WAS NOT GENERATED @
103663    123 042505 020113 EMS214: .ASCIZ @SEEK COMMAND @
103701    120 047522 051107 EMS215:  ASCIZ @PROGRAM TIMEOUT @
103722 052504 044522 043516 EMS216: .ASCIZ @DURING LOOK AHEAD TEST @
103752 047514 045517 040440 EMS217: .ASCIZ @LOOK AHEAD REGISTER,RMLA, @
104005    123 040505 041522 EMS220: .ASCIZ @SEARCH COMMAND @
104025    102 042101 051440 EMS221: .ASCIZ @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
104075    101 042040 052101 EMS222: .ASCIZ @A DATA TRANSFER COMMAND @
104126 042510 042101 051105 EMS223: .ASCIZ @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
104203    116 047117 054105 EMS224: .ASCIZ @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
104255    105  050130  052103  EM1:    .ASCIZ  @EXPCTD  RECEVD@
104274 052502  040523  051104  EM110:  .ASCIZ  @BUSADR@
104303    040  046522  051503  EM111:  .ASCIZ  @ RMCS2   RMCS1@

104322 042522  042503  042126  EM114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
104351    105  050130  052103  EM223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
104377    105  050130  052103  EM256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
104426 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

104454 042107  042101  051522  EM336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
104513    122  041515  031123  EM337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
104553    137  057537  057537          .ASCII  @                          @<CR><LF>
104613    105  050130  052103          .ASCIZ  @EXPCTD--RECEVD  --BIT---ADRESS@

104652 046522  051105  020061  EM344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
104713    137  057537  057537          .ASCII  @                  WORD     BIT@<CR><LF>
104752 054105  041520  042124          .ASCIZ  @EXPCTD--RECEVD  NUMBER  POSITON@
105012 054105  041520  042124  EM353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
105032 051040  046115  020101          .ASCIZ  @ RMLA    RMLA    RMOF @

105061    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
105126 020040  051040  040515          .ASCIZ  @   RMAS@
105136 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
105203    040  020040  051040          .ASCIZ  @    RMEC1   RMEC2@
105225    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
105263    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
```

```
                105322                          .EVEN
        105322  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
        105330  001276  000000          ED110:  .WORD   $BASE,0
        105334  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

        105342  001356  001176  001200  ED114:  .WORD   RMDT1,$TMP1,$TMP2,0
        105352  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

        105362  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

        105374  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
        105406  001140  001142  001432  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

        105416  001330  001340  001342  STSD1:  .WORD   RMCS11,RMCS21,RMDS1,RMER11,RMER21,RMAS1,0
        105434  001332  001334  001336  STSD2:  .WORD   RMWC1,RMBA1,RMDA1,RMOF1,RMDC1,RMEC11
        105450  001376  000000          .WORD   RMEC21,0
        105454  001336  001364  001362  STSD3:  .WORD   RMDA1,RMDC1,RMOF1,RMLA1,0
        105466  001354  001370  001356  STSD4:  .WORD   RMMR11,RMMR21,RMDT1,RMSN1,0
```

```
105500  000                 EF110:  .BYTE   0
105501  000  000            EF111:  .BYTE   0,0
105503  000  000  000       EF114:  .BYTE   0,0,0
105506  000  000  000       EF336:  .BYTE   0,0,0,0
105512  000  000  000       EF337:  .BYTE   0,0,0,0,0

105517  000  000  000       STSF:   .BYTE   0,0,0,0,0,0,0
```

```
                        ERROR MESSAGE STRINGS

                        .EVEN
105526  000402          MFGFIL: .BLKW   258.
106532  177777                  .WORD   -1
106534  177777                  .WORD   -1
106536  000402          USRFIL:         .BLKW   258.
107542  177777                  .WORD   -1
107544  177777                  .WORD   -1
                        .EVEN
```

```
107546                          BUFFER:
107546  000402                  BUFONE: .BLKW   258.
110552  000402                  BUFTWO: .BLKW   258.


        107546                  .       =       BUFFER
107546                          HELP:
107546  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>
107550  046011  051511  020124  .ASCII  @       LIST OF TESTS@<CR><LF>
107570  057411  057537  057537  .ASCII  @                       @<CR><LF>
107610  030524  041411  047117  .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
107643     124  004462  042504  .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
107675     124  004463  051104  .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
107721     124  004464  051127  .ASCII  @T4     WRITE, READ ZEROS@<CR><LF>
107747     124  004465  051127  .ASCII  @T5     WRITE, WRITE CHECK ZEROS@<CR><LF>
110004  033124  053411  044522  .ASCII  @T6     WRITE, WRITE CHECK ZEROS W/ WCE ERROR@<CR><LF>
110056  033524  053411  044522  .ASCII  @T7     WRITE, READ ONES@<CR><LF>
110103     124  030061  053411  .ASCII  @T10    WRITE, WRITE CHECK ONES@<CR><LF>
110140  030524  004461  051127  .ASCII  @T11    WRITE, WRITE CHECK ONES W/ WCE ERROR@<CR><LF>
110212  030524  004462  051127  .ASCII  @T12    WRITE, WRITE CHECK MULTIPLE SECTORS@<CR><LF>
110263     124  031461  053411  .ASCII  @T13    WRITE, READ W/ IMPLIED SEEK@<CR><LF>
110324  030524  004464  051127  .ASCII  @T14    WRITE, WRITE CHECK W/ HEAD SWITCHING@<CR><LF>
110376  030524  004465  051127  .ASCII  @T15    WRITE, WRITE CHECK W/ MID-TRANSFER SEEK@<CR><LF>
110453     124  033061  053411  .ASCII  @T16    WRITE, READ W/ HCE ERROR@<CR><LF>
110511     124  033461  053411  .ASCII  @T17    WRITE, READ W/ HCI@<CR><LF>
110541     124  030062  053411  .ASCII  @T20    WRITE, READ W/ IVC ERROR@<CR><LF>
110577     124  030462  053411  .ASCII  @T21    WRITE, READ W/ ABORT@<CR><LF>
110631     124  031062  053411  .ASCII  @T22    WRITE, READ W/ BAI@<CR><LF>
110661     124  031462  053411  .ASCII  @T23    WRITE, READ EACH CURRENT LEVEL@<CR><LF>
110725     124  032062  053411  .ASCII  @T24    WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING@<CR><LF>
111010  031124  004465  051127  .ASCII  @T25    WRITE, READ EARLY PEAK SHIFT@<CR><LF>
111052  031124  004466  051127  .ASCII  @T26    WRITE, READ MIXED PEAK SHIFT@<CR><LF>
111114  031124  004467  051127  .ASCII  @T27    WRITE, READ EACH TRACK@<CR><LF>
111150  005015                  .ASCII  <CR><LF>
111152  047411  042520  040522  .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
111210  057411  057537  057537  .ASCII  @       ------------------------@<CR><LF>
111246  005015                  .ASCII  <CR><LF>
111250  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
111266  026455  026455  026455  .ASCII  @------          --------------------@<CR><LF>
111324  020040  032461  004411  .ASCII  @  15           HALT ON ERROR@<CR><LF>
111351     040  030440  004464  .ASCII  @  14           LOOP ON TEST@<CR><LF>
111375     040  030440  004463  .ASCII  @  13           INHIBIT ERROR TYPEOUTS@<CR><LF>
111433     040  030440  004462  .ASCII  @  12           @<CR><LF>
111443     040  030440  004461  .ASCII  @  11           INHIBIT ITERATIONS@<CR><LF>
111475     040  030440  004460  .ASCII  @  10           BELL ON ERROR@<CR><LF>
111522  020040  034440  004411  .ASCII  @   9           LOOP ON ERROR@<CR><LF>
111547     040  020040  004470  .ASCII  @   8           LOOP ON TEST IN SWR<7:0>@<CR><LF>
111607     040  020040  004467  .ASCII  @   7           TN128@<CR><LF>
111624  020040  033040  004411  .ASCII  @   6           TN64@<CR><LF>
111640  020040  032440  004411  .ASCII  @   5           TN32@<CR><LF>
111654  020040  032040  004411  .ASCII  @   4           TN16@<CR><LF>
111670  020040  031440  004411  .ASCII  @   3           TN8@<CR><LF>
111703     040  020040  004462  .ASCII  @   2           TN4@<CR><LF>
```

```
111716  020040  030440  004411  .ASCII  @   1          TN2@<CR><LF>
111731     040  020040  004460  .ASCII  @   0          TN1@<CR><LF>
111744  005015                  .ASCII  <CR><LF>
111746  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1207#  1319   1360
ACDW1 = 000000          1319   1362
ACDW2 = 000000          1319   1363
ACKSTS  053676          9372   9560   11615#
ACPUOP= 000000          1319   1334
ADDW0 = 000000          1319   1364
ADDW1 = 000000          1319   1365
ADDW10= 000000          1319
ADDW11= 000000          1319
ADDW12= 000000          1319
ADDW13= 000000          1319
ADDW14= 000000          1319
ADDW15= 000000          1319
ADDW2 = 000000          1319   1366
ADDW3 = 000000          1319   1367
ADDW4 = 000000          1319   1368
ADDW5 = 000000          1319   1369
ADDW6 = 000000          1319   1370
ADDW7 = 000000          1319   1371
ADDW8 = 000000          1319
ADDW9 = 000000          1319
ADEVCT= 000000          1319   1325
ADEVM = 000000          1319   1361
ADR   = 000001          3483#  3484   3525#  3526#  3527   3569#  3570#  3571   3607#  3608#  3609   3864#  3865#
                        3866   4112#  4113#  4114   4404#  4405#  4406   4661#  4662#  4663   4909#  4910#  4911
                        5201#  5202#  5203   5454#  5455#  5456   5770#  5771#  5772   6023#  6024#  6025   6276#
                        6277#  6278   6687#  6688#  6689   7022#  7023#  7024   7260#  7261#  7262   7472#  7473#
                        7474   7737#  7738#  7739   8001#  8002#  8003   8249#  8250#  8251   8506#  8507#  8508
                        8763#  8764#  8765
AENV  = 000000          1319   1330
AENVM = 000000          1319   1331
AFATAL= 000000          1319   1322
AMADR1= 000000          1319   1347
AMADR2= 000000          1319   1351
AMADR3= 000000          1319   1354
AMADR4= 000000          1319   1357
AMAMS1= 000000          1319   1341
AMAMS2= 000000          1319   1349
AMAMS3= 000000          1319   1352
AMAMS4= 000000          1319   1355
AMSGAD= 000000          1319   1327
AMSGLG= 000000          1319   1328
AMSGTY= 000000          1319   1321
AMTYP1= 000000          1319   1342
AMTYP2= 000000          1319   1350
AMTYP3= 000000          1319   1353
AMTYP4= 000000          1319   1356
AOE   = 001000           987#   998   10785  10800  10815  10830  10834  11081  11085  13837
APASS = 000000          1319   1324
APE   = 100000          1186#
APRIOR= 000000          1319
APTCSU= 000040         13190  13835#
APTENV= 000001         13183  13369  13791  13833#
APTSIZ= 000200          3227  13832#
APTSPO= 000100         13185  13793  13834#
ARGS  = 000006          3620#  3638#  3660#  3668#  3671#  3674#  3682#  3702#  3726#  3734#  3737#  3740#  3748#
```

```
                3755#  3764#  3783#  3805#  3813#  3816#  3819#  3827#  3834#  3843#  3850#  3877#  3895#
                3917#  3925#  3928#  3931#  3939#  3959#  3983#  3991#  3994#  3997#  4005#  4012#  4021#
                4040#  4061#  4069#  4072#  4075#  4083#  4090#  4099#  4125#  4143#  4165#  4173#  4176#
                4179#  4187#  4207#  4231#  4239#  4242#  4245#  4253#  4260#  4269#  4291#  4312#  4320#
                4323#  4326#  4334#  4343#  4384#  4417#  4435#  4457#  4465#  4468#  4471#  4479#  4499#
                4523#  4531#  4534#  4537#  4545#  4552#  4561#  4580#  4602#  4610#  4613#  4616#  4624#
                4631#  4640#  4647#  4674#  4692#  4714#  4722#  4725#  4728#  4736#  4756#  4780#  4788#
                4791#  4794#  4802#  4809#  4818#  4837#  4858#  4866#  4869#  4872#  4880#  4887#  4896#
                4922#  4940#  4962#  4970#  4973#  4976#  4984#  5004#  5028#  5036#  5039#  5042#  5050#
                5057#  5066#  5088#  5109#  5117#  5120#  5123#  5131#  5140#  5181#  5214#  5232#  5254#
                5262#  5265#  5268#  5276#  5303#  5325#  5333#  5336#  5339#  5347#  5354#  5363#  5382#
                5403#  5411#  5414#  5417#  5425#  5432#  5441#  5467#  5485#  5507#  5515#  5518#  5521#
                5529#  5552#  5563#  5571#  5572#  5573#  5579#  5603#  5611#  5614#  5622#  5629#  5638#
                5660#  5671#  5679#  5680#  5681#  5687#  5710#  5719#  5722#  5730#  5737#  5746#  5753#
                5783#  5801#  5823#  5831#  5834#  5837#  5845#  5872#  5894#  5902#  5905#  5908#  5916#
                5923#  5932#  5951#  5972#  5980#  5983#  5986#  5994#  6001#  6010#  6036#  6054#  6076#
                6084#  6087#  6090#  6098#  6125#  6147#  6155#  6158#  6161#  6169#  6176#  6185#  6204#
                6225#  6233#  6236#  6239#  6247#  6254#  6263#  6293#  6311#  6340#  6348#  6351#  6354#
                6362#  6382#  6406#  6414#  6417#  6420#  6428#  6449#  6468#  6485#  6497#  6515#  6538#
                6546#  6549#  6552#  6560#  6580#  6599#  6616#  6628#  6657#  6702#  6720#  6749#  6757#
                6760#  6763#  6771#  6791#  6815#  6823#  6826#  6829#  6837#  6857#  6866#  6884#  6906#
                6914#  6917#  6920#  6928#  6948#  6957#  6964#  6993#  7038#  7053#  7095#  7103#  7106#
                7109#  7117#  7126#  7140#  7159#  7200#  7208#  7211#  7214#  7222#  7231#  7245#  7277#
                7291#  7325#  7326#  7332#  7346#  7349#  7357#  7365#  7384#  7416#  7418#  7426#  7441#
                7444#  7452#  7459#  7485#  7503#  7525#  7533#  7536#  7539#  7547#  7567#  7596#  7604#
                7607#  7610#  7618#  7625#  7634#  7653#  7678#  7686#  7689#  7692#  7700#  7707#  7716#
                7723#  7750#  7768#  7790#  7798#  7801#  7804#  7812#  7832#  7856#  7864#  7867#  7870#
                7878#  7885#  7894#  7913#  7936#  7944#  7947#  7950#  7958#  7965#  7974#  7981#  8014#
                8032#  8054#  8062#  8065#  8068#  8076#  8096#  8120#  8128#  8131#  8134#  8142#  8149#
                8158#  8177#  8198#  8206#  8209#  8212#  8220#  8227#  8236#  8262#  8280#  8302#  8310#
                8313#  8316#  8324#  8344#  8368#  8376#  8379#  8382#  8390#  8397#  8406#  8425#  8447#
                8455#  8458#  8461#  8469#  8476#  8485#  8492#  8519#  8537#  8559#  8567#  8570#  8573#
                8581#  8601#  8625#  8633#  8636#  8639#  8647#  8654#  8663#  8682#  8704#  8712#  8715#
                8718#  8726#  8733#  8742#  8749#  8776#  8794#  8816#  8824#  8827#  8830#  8838#  8858#
                8882#  8890#  8893#  8896#  8904#  8911#  8920#  8939#  8961#  8969#  8972#  8975#  8983#
                8990#  8999#  9006#
ASNDA   001500  1443#  6650   6987   9750*  9779   9825*  9826   9828*  9829*  9830   9832*  9877
ASNDC   001476  1442#  6649   6986   9749*  9778   9833*  9834   9836*  9876
ASWREG= 000000  1319   1332
ATA     = 100000   967#  10612  10613  10615  11347  11348  11383  11386  11878  11879  11918  11921  13837
ATESTN= 000000  1319   1323
ATNMSK= 000377  1004#  12075
ATNTBL  070774  3425   3443   13837#
AUNIT   = 000000  1319   1326
AUSWR   = 000000  1319   1333
AVECT1= 120254  1208#  1319   1358
AVECT2= 000000  1319   1359
A16     = 000400  1157#
A17     = 001000  1156#
BACK    = 000000  3620#  3625   3638#  3642   3660#  3664   3668#  3671#  3674#  3678   3682#  3686   3702#
                3707   3726#  3730   3734#  3737#  3740#  3744   3748#  3752   3755#  3759   3764#  3768
                3783#  3788   3805#  3809   3813#  3816#  3819#  3823   3827#  3831   3834#  3838   3843#
                3847   3850#  3856   3877#  3882   3895#  3899   3917#  3921   3925#  3928#  3931#  3935
                3939#  3943   3959#  3964   3983#  3987   3991#  3994#  3997#  4001   4005#  4009   4012#
                4016   4021#  4025   4040#  4045   4061#  4065   4069#  4072#  4075#  4079   4083#  4087
                4090#  4094   4099#  4103   4125#  4130   4143#  4147   4165#  4169   4173#  4176#  4179#
```

4183	4187#	4191	4207#	4212	4231#	4235	4239#	4242#	4245#	4249	4253#	4257
4260#	4264	4269#	4273	4291#	4296	4312#	4316	4320#	4323#	4326#	4330	4334#
4338	4343#	4347	4384#	4388	4417#	4422	4435#	4439	4457#	4461	4465#	4468#
4471#	4475	4479#	4483	4499#	4504	4523#	4527	4531#	4534#	4537#	4541	4545#
4549	4552#	4556	4561#	4565	4580#	4585	4602#	4606	4610#	4613#	4616#	4620
4624#	4628	4631#	4635	4640#	4644	4647#	4653	4674#	4679	4692#	4696	4714#
4718	4722#	4725#	4728#	4732	4736#	4740	4756#	4761	4780#	4784	4788#	4791#
4794#	4798	4802#	4806	4809#	4813	4818#	4822	4837#	4842	4858#	4862	4866#
4869#	4872#	4876	4880#	4884	4887#	4891	4896#	4900	4922#	4927	4940#	4944
4962#	4966	4970#	4973#	4976#	4980	4984#	4988	5004#	5009	5028#	5032	5036#
5039#	5042#	5046	5050#	5054	5057#	5061	5066#	5070	5088#	5093	5109#	5113
5117#	5120#	5123#	5127	5131#	5135	5140#	5144	5181#	5185	5214#	5219	5232#
5236	5254#	5258	5262#	5265#	5268#	5272	5276#	5280	5303#	5308	5325#	5329
5333#	5336#	5339#	5343	5347#	5351	5354#	5358	5363#	5367	5382#	5387	5403#
5407	5411#	5414#	5417#	5421	5425#	5429	5432#	5436	5441#	5445	5467#	5472
5485#	5489	5507#	5511	5515#	5518#	5521#	5525	5529#	5533	5552#	5557	5563#
5567	5571#	5572#	5573#	5577	5579#	5583	5603#	5607	5611#	5614#	5618	5622#
5626	5629#	5633	5638#	5642	5660#	5665	5671#	5675	5679#	5680#	5681#	5685
5687#	5691	5710#	5714	5719#	5722#	5726	5730#	5734	5737#	5741	5746#	5750
5753#	5759	5783#	5788	5801#	5805	5823#	5827	5831#	5834#	5837#	5841	5845#
5849	5872#	5877	5894#	5898	5902#	5905#	5908#	5912	5916#	5920	5923#	5927
5932#	5936	5951#	5956	5972#	5976	5980#	5983#	5986#	5990	5994#	5998	6001#
6005	6010#	6014	6036#	6041	6054#	6058	6076#	6080	6084#	6087#	6090#	6094
6098#	6102	6125#	6130	6147#	6151	6155#	6158#	6161#	6165	6169#	6173	6176#
6180	6185#	6189	6204#	6209	6225#	6229	6233#	6236#	6239#	6243	6247#	6251
6254#	6258	6263#	6267	6293#	6298	6311#	6315	6340#	6344	6348#	6351#	6354#
6358	6362#	6366	6382#	6387	6406#	6410	6414#	6417#	6420#	6424	6428#	6432
6449#	6453	6468#	6472	6485#	6489	6497#	6501	6515#	6520	6538#	6542	6546#
6549#	6552#	6556	6560#	6564	6580#	6584	6599#	6603	6616#	6620	6628#	6632
6657#	6662	6702#	6707	6720#	6724	6749#	6753	6757#	6760#	6763#	6767	6771#
6775	6791#	6796	6815#	6819	6823#	6826#	6829#	6833	6837#	6841	6857#	6861
6866#	6870	6884#	6889	6906#	6910	6914#	6917#	6920#	6924	6928#	6932	6948#
6952	6957#	6961	6964#	6970	6993#	6998	7038#	7053#	7058	7095#	7099	7103#
7106#	7109#	7113	7117#	7121	7126#	7130	7140#	7144	7159#	7164	7200#	7204
7208#	7211#	7214#	7218	7222#	7226	7231#	7235	7245#	7249	7277#	7291#	7296
7325#	7326#	7330	7332#	7336	7346#	7349#	7353	7357#	7361	7365#	7369	7384#
7389	7416#	7418#	7422	7426#	7430	7441#	7444#	7448	7452#	7456	7459#	7463
7485#	7490	7503#	7507	7525#	7529	7533#	7536#	7539#	7543	7547#	7551	7567#
7572	7596#	7600	7604#	7607#	7610#	7614	7618#	7622	7625#	7629	7634#	7638
7653#	7658	7678#	7682	7686#	7689#	7692#	7696	7700#	7704	7707#	7711	7716#
7720	7723#	7729	7750#	7755	7768#	7772	7790#	7794	7798#	7801#	7804#	7808
7812#	7816	7832#	7837	7856#	7860	7864#	7867#	7870#	7874	7878#	7882	7885#
7889	7894#	7898	7913#	7918	7936#	7940	7944#	7947#	7950#	7954	7958#	7962
7965#	7969	7974#	7978	7981#	7987	8014#	8019	8032#	8036	8054#	8058	8062#
8065#	8068#	8072	8076#	8080	8096#	8101	8120#	8124	8128#	8131#	8134#	8138
8142#	8146	8149#	8153	8158#	8162	8177#	8182	8198#	8202	8206#	8209#	8212#
8216	8220#	8224	8227#	8231	8236#	8240	8262#	8267	8280#	8284	8302#	8306
8310#	8313#	8316#	8320	8324#	8328	8344#	8349	8368#	8372	8376#	8379#	8382#
8386	8390#	8394	8397#	8401	8406#	8410	8425#	8430	8447#	8451	8455#	8458#
8461#	8465	8469#	8473	8476#	8480	8485#	8489	8492#	8498	8519#	8524	8537#
8541	8559#	8563	8567#	8570#	8573#	8577	8581#	8585	8601#	8606	8625#	8629
8633#	8636#	8639#	8643	8647#	8651	8654#	8658	8663#	8667	8682#	8687	8704#
8708	8712#	8715#	8718#	8722	8726#	8730	8733#	8737	8742#	8746	8749#	8755
8776#	8781	8794#	8798	8816#	8820	8824#	8827#	8830#	8834	8838#	8842	8858#
8863	8882#	8886	8890#	8893#	8896#	8900	8904#	8908	8911#	8915	8920#	8924
8939#	8944	8961#	8965	8969#	8972#	8975#	8979	8983#	8987	8990#	8994	8999#

```
                        9003    9006#   9012
BADSCT  040734          3638    3895    4143    4435    4692    4940    5232    5485    5801    6054    6311    6720    7503
                        7768    8032    8280    8537    8794    9466#
BA1    = 000010         1176#   7581    11001
BB00   = 000001         1094#
BB01   = 000002         1093#
BB02   = 000004         1092#
BB03   = 000010         1091#
BB04   = 000020         1090#
BB05   = 000040         1089#
BB06   = 000100         1088#
BB07   = 000200         1087#
BB08   = 000400         1086#
BB09   = 001000         1085#
BIT0   = 000001          883#   3272    3593    3595    10288
BIT00  = 000001          873#    883     909     958     977     996    1034    1053    1094    1180    1196
BIT01  = 000002          872#    882     908     957     995    1033    1052    1093    1179    1195
BIT02  = 000004          871#    881     907     956     994    1032    1051    1092    1178    1194
BIT03  = 000010          870#    880     906     955     993    1031    1050    1091    1106    1176    1193
BIT04  = 000020          869#    879     905     954     992    1049    1090    1175
BIT05  = 000040          868#    878     904     991    1030    1048    1089    1174
BIT06  = 000100          867#    877     976     990    1012    1029    1047    1088    1159    1173    1192
BIT07  = 000200          866#    876     975     989    1011    1028    1046    1070    1087    1105    1158    1172
BIT08  = 000400          865#    875     953     974     988    1010    1027    1045    1086    1157    1170   13270
BIT09  = 001000          864#    874     952     973     987    1009    1026    1044    1085    1156    1169   13288   13380
BIT1   = 000002          882#   9756    9914   11018
BIT10  = 002000          863#    951     972     986    1008    1025    1043    1069    1084    1104    1155    1168    1191
                        13357
BIT11  = 004000          862#    903     971     985    1024    1042    1060    1068    1083    1103    1167    1190    9384
                        13295
BIT12  = 010000          861#    970     984    1023    1041    1067    1082    1102    1166    1189    9334
BIT13  = 020000          860#    969     983    1022    1040    1059    1081    1101    1153    1165    1188    6289    6643
                        13364
BIT14  = 040000          859#    968     982    1021    1039    1058    1080    1100    1111    1152    1164    1187    9299
                        13256
BIT15  = 100000          858#    967     981    1020    1038    1057    1079    1099    1110    1151    1163    1186    9286
                         9999   10053
BIT2   = 000004          881#   10288
BIT3   = 000010          880#   9412    12460   12547
BIT4   = 000020          879#   9362    12566
BIT5   = 000040          878#
BIT6   = 000100          877#   9311
BIT7   = 000200          876#   3267    10294
BIT8   = 000400          875#
BIT9   = 001000          874#
BOTADR  037756          9150*   9168*   9171    9186    9231#
BOTFLG  037760          9136*   9178*   9181    9184*   9232#
BPTVEC= 000014           890#
BSE    = 100000         1099#   6482    6493    6613    6624    6849    6852    6940    6943    9662   10905
BUFFER  107546          9481*   9730   13837#
BUFONE  107546          3631    3714    3851    3888    3971    4136    4220    4428    4511    4648    4685    4768    4933
                        5017    5225    5289    5478    5592    5754    5794    5858    6047    6112    6304    6322    6324*
                        6326*   6394    6651    6713    6731    6733*   6735*   6805    6965    6988    7032    7085    7271
                        7496    7584    7724    7761    7845    7982    8025    8109    8273    8357    8493    8530    8614
                        8750    8787    8870    9007   13837#
BUFTWO  110552          3794    3852    4288*   4354    4369    4371    4392*   4591    4649    5085*   5151    5166    5168
```

```
                    5189*  5700   5755   6526   6896   6966   7190   7667   7725   7924   7983   8437   8494
                    8693   8751   8950   9008   13837#
CC     = 004000     1083#
CH     = 002000     1084#
CLKCNT  037761      9137*  9155*  9161*  9162   9165*  9169   9174*  9185*  9233#
CKSWR  = 104410     13254  13353  13379  13730#
CLEAN  = ****** U   6287   6641   13837
CLKADR  001502      1444#  10250* 10252  10256* 10258  10283
CLKVCT  001504      1445#  10251* 10257*
CLR    = 000040     1174#  3274   11444
CLRSTS  053016      9321   11480#
CLSPRN  067574      3345   3361   3382   13837#
CMNSTA  006746      3297   3432#
CMPBUF  043104      3850   4647   5753   6964   7723   7981   8492   8749   9006   9979#
CMPERR  063142      12859#
CNSL00  070072      3307   13837#
CNSL01  070131      3342   13837#
CNSL02  070156      3351   13837#
CNSL03  070237      3356   13837#
CNSL04  070267      3367   13837#
CNSL05  070343      3377   13837#
CNSL06  070377      3388   13837#
CNSL07  070424      3401   13837#
CNTCLR  052700      3532   3576   9301   11436#
CONT   = 000100     1047#
CR     = 000015      798#  3415   9153   12942  13229  13239  13837
CRLF   = 000200      799#  3251   13200  13239
CTLFG   001326      1374#  3471*  12931* 12933
CYLMSK= 001777      1074#  9917
DBCK   = 100000     1020#
DBEN   = 040000     1021#
DBL    = 002000     1191#
DCK    = 100000      981#   998   9655   9987   10943  10963  12553  12556
DDISP  = 177570      805#  1291   3215
DEBL   = 020000     1022#
DEVSEL  051226      9288   11146#
DISPLA  001156      1291#  3215*  3223*  13310* 13356*
DISPRE  000174      1218#  3223
DLT    = 100000     1163#  10787  12632  12635
DMD    = 000001     1034#  1053#  7064   7170
DPE    = 000010     1106#  9656   9662   10449  10729  11225  11764  12185  12606  12609
DPEHI  = 040000     1187#
DPELO  = 020000     1188#
DPR    = 000400      974#  11386  11583  11584  12050
DRQ    = 004000     1060#
DRVCLR= 000010       918#  9509   12038
DRVSTS  056234      9527   12029#
DRY    = 000200      975#  10416  10512  10515  10516  11386  11583  11584  12050
DSWR   = 177570      804#  1290   3214
DTASTS  057036      3682   3755   3834   3939   4012   4090   4187   4260   4343   4479   4552   4631   4736
                    4809   4887   4984   5057   5140   5276   5354   5432   5529   5629   5737   5845   5923
                    6001   6098   6176   6254   6362   6449   6468   6485   6580   6599   6616   6771   6857
                    6948   7126   7231   7547   7625   7707   7812   7885   7965   8076   8149   8227   8324
                    8397   8476   8581   8654   8733   8838   8911   8990   9630   12160#
DTE    = 010000      984#   998   12419  12437  12440
DTO    = 010000     1023#
```

```
DULPRT= 024024  1063#  3591  3595  3598
DVA   = 004000   903#  3280  3551 10139 10212 10377 10380 11171 11489 12037 12038
DVC   = 000200  1105# 11318 11573 11825 11863 11866 11999 12372 12376 12379 12424 12798 12803
                12806
EARLY   071320  8286 13837#
EBL   = 020000  1040#
ECH   = 000100   990#   998  9655  9989 10926 10941 10959 10965 12563 13837
ECI   = 004000  1068#  9991 10961 12560
ECRC  = 001000  1044#
EDT1    075650  1479  1486  1493  1500  1507  1514  1528  1535  1542  1549  1556  1563  1570
                1577  1584  1591  1598  1605  1612  1619  1626  1633  1640  1647  1654  1661
                1668  1675  1682  1689  1696  1703  1710  1717  1724  1731  1738  1745  1752
                1759  1766  1774  1781  1788  1795  1802  1810  1818  1826  1840  1847  1854
                1861  1868  1875  1882  1890  1897  1904  1911  1918  1925  1932  1939  1946
                1953  1960  1967  1974  2016  2023  2030  2037  2044  2051  2058  2065  2072
                2079  2086  2093  2100  2107  2114  2121  2128  2135  2142  2149  2156  2163
                2170  2177  2184  2191  2198  2205  2212  2219  2226  2233  2240  2247  2254
                2261  2268  2275  2290  2297  2304  2311  2325  2332  2339  2346  2353  2360
                2367  2374  2381  2388  2395  2402  2409  2416  2423  2430  2437  2444  2451
                2472  2479  2486  2493  2500  2640  2647  2654  2675  2682  2689  2703  2710
                2717  2724  2731  2738  2745  2752  2760  2767  2775  2782  2789  2797  2805
                2813  2822  2830  2838  2845  2852  2859  2866  2873  2880  2887  2894  2901
                2908  2915  2922  2929  2936  2943  2950  2957  2964  2971  2978  2985  2992
                2999  3006  3013  3020  3027  3034  3069  3076  3090  3097  3104  3111  3118
                3139 13837#
EDT110  075666  1981 13837#
EDT111  075670  1988  1995 13837#
EDT114  075672  2009 13837#
EDT2    075660  2458  2465  2661  2668 13837#
EDT223  075674  2507  2696 13837#
EDT336  075704  2283  3041  3055  3062 13837#
EDT337  075714  3048 13837#
EDT344  075724  3083 13837#
EDT353  075736  3132 13837#
ED1     105322 13837#
ED110   105330 13837#
ED111   105334 13837#
ED114   105342 13837#
ED223   105352 13837#
ED336   105362 13837#
ED337   105374 13837#
ED353   105406 13837#
EECC  = 000020  1049#
EFT1    075740  1480  1487  1494  1501  1508  1515  1529  1536  1543  1550  1557  1564  1571
                1578  1585  1592  1599  1606  1613  1620  1627  1634  1641  1648  1655  1662
                1669  1676  1683  1690  1697  1704  1711  1718  1725  1732  1739  1746  1753
                1760  1767  1775  1782  1789  1796  1803  1811  1819  1827  1841  1848  1855
                1862  1869  1876  1883  1891  1898  1905  1912  1919  1926  1933  1940  1947
                1954  1961  1968  1975  2017  2024  2031  2038  2045  2052  2059  2066  2073
                2080  2087  2094  2101  2108  2115  2122  2129  2136  2143  2150  2157  2164
                2171  2178  2185  2192  2199  2206  2213  2220  2227  2234  2241  2248  2255
                2262  2269  2276  2291  2298  2305  2312  2326  2333  2340  2347  2354  2361
                2368  2375  2382  2389  2396  2403  2410  2417  2424  2431  2438  2445  2452
                2473  2480  2487  2494  2501  2641  2648  2655  2676  2683  2690  2704  2711
                2718  2725  2732  2739  2746  2753  2761  2768  2776  2783  2790  2798  2806
                2814  2823  2831  2839  2846  2853  2860  2867  2874  2881  2888  2895  2902
```

		2909	2916	2923	2930	2937	2944	2951	2958	2965	2972	2979	2986	2993
		3000	3007	3014	3021	3028	3035	3070	3077	3091	3098	3105	3112	3119
		3140	13837#											
EFT110	075756	1982	13837#											
EFT111	075760	1989	1996	13837#										
EFT114	075762	2010	13837#											
EFT2	075750	2459	2466	2662	2669	13837#								
EFT223	075764	2508	2697	13837#										
EFT336	075774	2284	3042	3056	3063	13837#								
EFT337	076004	3049	13837#											
EFT344	076014	3084	13837#											
EFT353	076024	3133	13837#											
EF110	105500	13837#												
EF111	105501	13837#												
EF114	105503	13837#												
EF336	105506	13837#												
EF337	105512	13837#												
EMT1	075524	1478	1485	1492	1499	1506	1513	1527	1534	1541	1548	1555	1562	1569
		1576	1583	1590	1597	1604	1611	1618	1625	1632	1639	1646	1653	1660
		1667	1674	1681	1688	1695	1702	1709	1716	1723	1730	1737	1744	1751
		1758	1765	1773	1780	1787	1794	1801	1809	1817	1825	1839	1846	1853
		1860	1867	1874	1881	1889	1896	1903	1910	1917	1924	1931	1938	1945
		1952	1959	1966	1973	2015	2022	2029	2036	2043	2050	2057	2064	2071
		2078	2085	2092	2099	2106	2113	2120	2127	2134	2141	2148	2155	2162
		2169	2176	2183	2190	2197	2204	2211	2218	2225	2232	2239	2246	2253
		2260	2267	2274	2289	2296	2303	2310	2324	2331	2338	2345	2352	2359
		2366	2373	2380	2387	2394	2401	2408	2415	2422	2429	2436	2443	2450
		2471	2478	2485	2492	2499	2639	2646	2653	2674	2681	2688	2702	2709
		2716	2723	2730	2737	2744	2751	2759	2766	2774	2781	2788	2796	2804
		2812	2821	2829	2837	2844	2851	2858	2865	2872	2879	2886	2893	2900
		2907	2914	2921	2928	2935	2942	2949	2956	2963	2970	2977	2984	2991
		2998	3005	3012	3019	3026	3033	3068	3075	3089	3096	3103	3110	3117
		3138	13837#											
EMT110	075546	1980	13837#											
EMT111	075552	1987	1994	13837#										
EMT114	075556	2008	13837#											
EMT2	075536	2457	2464	2660	2667	13837#								
EMT223	075562	2506	13837#											
EMT256	075574	2695	13837#											
EMT336	075606	2282	3040	3054	3061	13837#								
EMT337	075620	3047	13837#											
EMT344	075632	3082	13837#											
EMT353	075644	3131	13837#											
EM1	104255	13837#												
EM110	104274	13837#												
EM111	104303	13837#												
EM114	104322	13837#												
EM223	104351	13837#												
EM256	104377	13837#												
EM336	104454	13837#												
EM337	104513	13837#												
EM344	104652	13837#												
EM353	105012	13837#												
EMS1	076026	13837#												
EMS10	076362	13837#												
EMS100	100602	13837#												

EMS101	100630	13837#
EMS102	100656	13837#
EMS103	100705	13837#
EMS104	100714	13837#
EMS105	100743	13837#
EMS106	100772	13837#
EMS11	076424	13837#
EMS110	101011	13837#
EMS111	101040	13837#
EMS112	101051	13837#
EMS113	101135	13837#
EMS114	101165	13837#
EMS115	101211	13837#
EMS116	101216	13837#
EMS117	101246	13837#
EMS12	076435	13837#
EMS120	101270	13837#
EMS121	101305	13837#
EMS122	101350	13837#
EMS123	101405	13837#
EMS124	101450	13837#
EMS125	101502	13837#
EMS126	101545	13837#
EMS127	101610	13837#
EMS13	076476	13837#
EMS130	101647	13837#
EMS131	101705	13837#
EMS132	101745	13837#
EMS133	101771	13837#
EMS134	102014	13837#
EMS135	102056	13837#
EMS136	102120	13837#
EMS137	102145	13837#
EMS14	076507	13837#
EMS140	102202	13837#
EMS141	102223	13837#
EMS142	102251	13837#
EMS143	102266	13837#
EMS144	102314	13837#
EMS145	102324	13837#
EMS146	102353	13837#
EMS147	102372	13837#
EMS15	076520	13837#
EMS150	102407	13837#
EMS151	102426	13837#
EMS152	102452	13837#
EMS153	102476	13837#
EMS154	102520	13837#
EMS155	102534	13837#
EMS156	102554	13837#
EMS157	102572	13837#
EMS16	076542	13837#
EMS160	102607	13837#
EMS161	102624	13837#
EMS162	102654	13837#
EMS163	102707	13837#

EMS164	102733	13837#
EMS165	103014	13837#
EMS166	103030	13837#
EMS167	103055	13837#
EMS17	076572	13837#
EMS170	103102	13837#
EMS171	103131	13837#
EMS172	103164	13837#
EMS173	103204	13837#
EMS174	103221	13837#
EMS175	103227	13837#
EMS176	103255	13837#
EMS177	103304	13837#
EMS2	076075	13837#
EMS20	076632	13837#
EMS200	103326	13837#
EMS201	103343	13837#
EMS202	103411	13837#
EMS203	103441	13837#
EMS204	103454	13837#
EMS205	103502	13837#
EMS206	103514	13837#
EMS207	103541	13837#
EMS21	076655	13837#
EMS210	103566	13837#
EMS211	103577	13837#
EMS212	103615	13837#
EMS213	103640	13837#
EMS214	103663	13837#
EMS215	103701	13837#
EMS216	103722	13837#
EMS217	103752	13837#
EMS22	076700	13837#
EMS220	104005	13837#
EMS221	104025	13837#
EMS222	104075	13837#
EMS223	104126	13837#
EMS224	104203	13837#
EMS23	076714	13837#
EMS24	076742	13837#
EMS25	076771	13837#
EMS26	077006	13837#
EMS27	077027	13837#
EMS3	076112	13837#
EMS30	077040	13837#
EMS31	077050	13837#
EMS32	077077	13837#
EMS33	077126	13837#
EMS34	077154	13837#
EMS35	077225	13837#
EMS36	077254	13837#
EMS37	077303	13837#
EMS4	076155	13837#
EMS40	077331	13837#
EMS41	077360	13837#
EMS42	077406	13837#

EMS43	077435	13837#		
EMS44	077464	13837#		
EMS45	077537	13837#		
EMS46	077602	13837#		
EMS47	077631	13837#		
EMS5	076220	13837#		
EMS50	077660	13837#		
EMS51	077707	13837#		
EMS52	077735	13837#		
EMS53	077747	13837#		
EMS54	077761	13837#		
EMS55	100003	13837#		
EMS56	100032	13837#		
EMS57	100107	13837#		
EMS6	076250	13837#		
EMS60	100135	13837#		
EMS61	100150	13837#		
EMS62	100177	13837#		
EMS63	100215	13837#		
EMS64	100243	13837#		
EMS65	100261	13837#		
EMS66	100327	13837#		
EMS67	100377	13837#		
EMS7	076315	13837#		
EMS70	100424	13837#		
EMS71	100436	13837#		
EMS72	100451	13837#		
EMS73	100461	13837#		
EMS74	100476	13837#		
EMS75	100515	13837#		
EMS76	100523	13837#		
EMS77	100554	13837#		
EMTVEC=	000030	893#	3198*	3199*
EMT1	071432	1477	13837#	
EMT10	071502	1526	13837#	
EMT100	072500	1923	13837#	
EMT101	072516	1930	13837#	
EMT102	072534	1937	13837#	
EMT103	072544	1944	13837#	
EMT104	072556	1951	13837#	
EMT105	072574	1958	13837#	
EMT106	072604	1965	13837#	
EMT107	072614	1972	13837#	
EMT11	071506	1533	13837#	
EMT110	072634	1979	13837#	
EMT111	072646	1986	13837#	
EMT112	072654	1993	13837#	
EMT113	072662	2000	13837#	
EMT114	072676	2007	13837#	
EMT115	072704	2014	13837#	
EMT116	072714	2021	13837#	
EMT117	072724	2028	13837#	
EMT12	071512	1540	13837#	
EMT120	072734	2035	13837#	
EMT121	072744	2042	13837#	
EMT122	072754	2049	13837#	

```
EMT123  072764            2056   13837#
EMT124  072774            2063   13837#
EMT125  073004            2070   13837#
EMT126  073014            2077   13837#
EMT127  073026            2084   13837#
EMT13   071520            1547   13837#
EMT130  073040            2091   13837#
EMT131  073052            2098   13837#
EMT132  073064            2105   13837#
EMT133  073076            2112   13837#
EMT134  073110            2119   13837#
EMT135  073122            2126   13837#
EMT136  073134            2133   13837#
EMT137  073146            2140   13837#
EMT14   071532            1554   13837#
EMT140  073156            2147   13837#
EMT141  073166            2154   13837#
EMT142  073176            2161   13837#
EMT143  073206            2168   13837#
EMT144  073216            2175   13837#
EMT145  073226            2182   13837#
EMT146  073236            2189   13837#
EMT147  073246            2196   13837#
EMT15   071540            1561   13837#
EMT150  073256            2203   13837#
EMT151  073266            2210   13837#
EMT152  073300            2217   13837#
EMT153  073312            2224   13837#
EMT154  073330            2231   13837#
EMT155  073346            2238   13837#
EMT156  073360            2245   13837#
EMT157  073372            2252   13837#
EMT16   071546            1568   13837#
EMT160  073404            2259   13837#
EMT161  073414            2266   13837#
EMT162  073426            2273   13837#
EMT163  073440          13837#
EMT164  073450            2288   13837#
EMT165  073456            2295   13837#
EMT166  073464            2302   13837#
EMT167  073472            2309   13837#
EMT17   071556            1575   13837#
EMT170  073500          13837#
EMT171  073510            2323   13837#
EMT172  073520            2330   13837#
EMT173  073530            2337   13837#
EMT174  073540            2344   13837#
EMT175  073552            2351   13837#
EMT176  073560            2358   13837#
EMT177  073566            2365   13837#
EMT2    071436            1484   13837#
EMT20   071566            1582   13837#
EMT200  073600            2372   13837#
EMT201  073612            2379   13837#
EMT202  073624            2386   13837#
EMT203  073636            2393   13837#
```

EMT204	073650	2400	13837#
EMT205	073666	2407	13837#
EMT206	073676	2414	13837#
EMT207	073706	2421	13837#
EMT21	071576	1589	13837#
EMT210	073720	2428	13837#
EMT211	073726	2435	13837#
EMT212	073742	2442	13837#
EMT213	073756	2449	13837#
EMT214	073766	2456	13837#
EMT215	074006	2463	13837#
EMT216	074020	2470	13837#
EMT217	074030	2477	13837#
EMT22	071606	1596	13837#
EMT220	074040	2484	13837#
EMT221	074050	2491	13837#
EMT222	074060	2498	13837#
EMT223	074070	2505	13837#
EMT224	074100	13837#	
EMT225	074102	13837#	
EMT226	074104	13837#	
EMT227	074106	13837#	
EMT23	071616	1603	13837#
EMT230	074110	13837#	
EMT231	074112	13837#	
EMT232	074114	13837#	
EMT233	074116	13837#	
EMT234	074120	13837#	
EMT235	074122	13837#	
EMT236	074124	13837#	
EMT237	074126	13837#	
EMT24	071626	1610	13837#
EMT240	074130	13837#	
EMT241	074132	13837#	
EMT242	074134	13837#	
EMT243	074136	13837#	
EMT244	074140	13837#	
EMT245	074142	13837#	
EMT246	074144	2638	13837#
EMT247	074154	2645	13837#
EMT25	071636	1617	13837#
EMT250	074166	2652	13837#
EMT251	074202	2659	13837#
EMT252	074210	2666	13837#
EMT253	074216	2673	13837#
EMT254	074234	2680	13837#
EMT255	074244	2687	13837#
EMT256	074254	2694	13837#
EMT257	074264	2701	13837#
EMT26	071646	1624	13837#
EMT260	074276	2708	13837#
EMT261	074310	2715	13837#
EMT262	074330	2722	13837#
EMT263	074340	2729	13837#
EMT264	074350	2736	13837#
EMT265	074370	2743	13837#

EMT266	074410	2750	13837#	
EMT267	074422	2758	13837#	
EMT27	071656	1631	13837#	
EMT270	074440	2765	13837#	
EMT271	074454	2773	13837#	
EMT272	074472	2780	13837#	
EMT273	074510	2787	13837#	
EMT274	074526	2795	13837#	
EMT275	074544	2803	13837#	
EMT276	074556	2811	13837#	
EMT277	074572	2820	13837#	
EMT3	071444	1491	13837#	
EMT30	071666	1638	13837#	
EMT300	074610	2828	13837#	
EMT301	074630	2836	13837#	
EMT302	074652	2843	13837#	
EMT303	074664	2850	13837#	
EMT304	074676	2857	13837#	
EMT305	074710	2864	13837#	
EMT306	074722	2871	13837#	
EMT307	074732	2878	13837#	
EMT31	071676	1645	13837#	
EMT310	074752	2885	13837#	
EMT311	074764	2892	13837#	
EMT312	074776	2899	13837#	
EMT313	075010	2906	13837#	
EMT314	075030	2913	13837#	
EMT315	075042	2920	13837#	
EMT316	075054	2927	13837#	
EMT317	075066	2934	13837#	
EMT32	071706	1652	13837#	
EMT320	075076	2941	13837#	
EMT321	075106	2948	13837#	
EMT322	075116	2955	13837#	
EMT323	075124	2962	13837#	
EMT324	075134	2969	13837#	
EMT325	075146	2976	13837#	
EMT326	075160	2983	13837#	
EMT327	075176	2990	13837#	
EMT33	071716	1659	13837#	
EMT330	075210	2997	13837#	
EMT331	075220	3004	13837#	
EMT332	075232	3011	13837#	
EMT333	075244	3018	13837#	
EMT334	075262	3025	13837#	
EMT335	075300	3032	13837#	
EMT336	075320	2281	3039	13837#
EMT337	075330	3046	13837#	
EMT34	071726	1666	13837#	
EMT340	075340	3053	13837#	
EMT341	075352	3060	13837#	
EMT342	075360	3067	13837#	
EMT343	075372	3074	13837#	
EMT344	075404	3081	13837#	
EMT345	075416	3088	13837#	
EMT346	075426	3095	13837#	

```
EMT347  075442            3102   13837#
EMT35   071736            1673   13837#
EMT350  075454            3109   13837#
EMT351  075472            3116   13837#
EMT352  075502            3123   13837#
EMT353  075506            3130   13837#
EMT354  075516            3137   13837#
EMT36   071746            1680   13837#
EMT37   071756            1687   13837#
EMT4    071452            1498   13837#
EMT40   071766            1694   13837#
EMT41   071774            1701   13837#
EMT42   072004            1708   13837#
EMT43   072016            1715   13837#
EMT44   072026            1722   13837#
EMT45   072036            1729   13837#
EMT46   072046            1736   13837#
EMT47   072056            1743   13837#
EMT5    071460            1505   13837#
EMT50   072064            1750   13837#
EMT51   072074            1757   13837#
EMT52   072106            1764   13837#
EMT53   072124            1772   13837#
EMT54   072142            1779   13837#
EMT55   072152            1786   13837#
EMT56   072164            1793   13837#
EMT57   072202            1800   13837#
EMT6    071466            1512   13837#
EMT60   072212            1808   13837#
EMT61   072230            1816   13837#
EMT62   072250            1824   13837#
EMT63   072264           13837#
EMT64   072266            1838   13837#
EMT65   072306            1845   13837#
EMT66   072326            1852   13837#
EMT67   072340            1859   13837#
EMT7    071474            1519   13837#
EMT70   072352            1866   13837#
EMT71   072362            1873   13837#
EMT72   072372            1880   13837#
EMT73   072410            1888   13837#
EMT74   072426            1895   13837#
EMT75   072436            1902   13837#
EMT76   072456            1909   13837#
EMT77   072466            1916   13837#
ENRGDT  071432           13837#
EQUALS  067575           13837#
ERR   = 040000             968#  10558   10560   10576   10610
ERRNMB  037754            9117*   9118*   9121    9133    9138    9230#
ERRTYP  037216            9096#  13366
ERRVEC= 000004             886#   3212    3213*   3224*   3492    3493    3494*   3495*   3506*   3507*   3511*   3512*   3538
                          3539    3540*   3541*   3545*   3546*   3557*   3558*  10130   10131   10135*  10136*  10167*  10168*
                         10203   10204   10208*  10209*  10234*  10235*  10246   10247   10248*  10249*  10255*  10263*  10264*
                         10278   10279   10280*  10281*  10305*  10306*  11153   11154   11157*  11158*  11191*  11192*  11439
                         11440   11441*  11442*  11453*  11454*  13261   13262*  13264*  13267*
ERTY00  037762            9103    9235#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
ERTY01  037770          9111   9236#
ERTY02  040000          9120   9238#
ERTY03  040007          9126   9240#
ERTY04  040015          9222   9242#
ESRC   = 004000         1042#
FER    = 000020          992#    998   6465   6475   6596   6606   6844   6847   6935   6938   9655  10890  12488
                       12505  12508  12531  12534
FIND   = 000001         3620#  3622#  3638#  3639   3660#  3661   3668#  3671#  3674#  3675   3682#  3683   3702#
                        3704#  3726#  3727   3734#  3737#  3740#  3741   3748#  3749   3755#  3756   3764#  3765
                        3783#  3785#  3805#  3806   3813#  3816#  3819#  3820   3827#  3828   3834#  3835   3843#
                        3844   3850#  3853   3877#  3879#  3895#  3896   3917#  3918   3925#  3928#  3931#  3932
                        3939#  3940   3959#  3961#  3983#  3984   3991#  3994#  3997#  3998   4005#  4006   4012#
                        4013   4021#  4022   4040#  4042#  4061#  4062   4069#  4072#  4075#  4076   4083#  4084
                        4090#  4091   4099#  4100   4125#  4127#  4143#  4144   4165#  4166   4173#  4176#  4179#
                        4180   4187#  4188   4207#  4209#  4231#  4232   4239#  4242#  4245#  4246   4253#  4254
                        4260#  4261   4269#  4270   4291#  4293#  4312#  4313   4320#  4323#  4326#  4327   4334#
                        4335   4343#  4344   4384#  4385   4417#  4419#  4435#  4436   4457#  4458   4465#  4468#
                        4471#  4472   4479#  4480   4499#  4501#  4523#  4524   4531#  4534#  4537#  4538   4545#
                        4546   4552#  4553   4561#  4562   4580#  4582#  4602#  4603   4610#  4613#  4616#  4617
                        4624#  4625   4631#  4632   4640#  4641   4647#  4650   4674#  4676#  4692#  4693   4714#
                        4715   4722#  4725#  4728#  4729   4736#  4737   4756#  4758#  4780#  4781   4788#  4791#
                        4794#  4795   4802#  4803   4809#  4810   4818#  4819   4837#  4839#  4858#  4859   4866#
                        4869#  4872#  4873   4880#  4881   4887#  4888   4896#  4897   4922#  4924#  4940#  4941
                        4962#  4963   4970#  4973#  4976#  4977   4984#  4985   5004#  5006#  5028#  5029   5036#
                        5039#  5042#  5043   5050#  5051   5057#  5058   5066#  5067   5088#  5090#  5109#  5110
                        5117#  5120#  5123#  5124   5131#  5132   5140#  5141   5181#  5182   5214#  5216#  5232#
                        5233   5254#  5255   5262#  5265#  5268#  5269   5276#  5277   5303#  5305#  5325#  5326
                        5333#  5336#  5339#  5340   5347#  5348   5354#  5355   5363#  5364   5382#  5384#  5403#
                        5404   5411#  5414#  5417#  5418   5425#  5426   5432#  5433   5441#  5442   5467#  5469#
                        5485#  5486   5507#  5508   5515#  5518#  5521#  5522   5529#  5530   5552#  5554#  5563#
                        5564   5571#  5572#  5573#  5574   5579#  5580   5603#  5604   5611#  5614#  5615   5622#
                        5623   5629#  5630   5638#  5639   5660#  5662#  5671#  5672   5679#  5680#  5681#  5682
                        5687#  5688   5710#  5711   5719#  5722#  5723   5730#  5731   5737#  5738   5746#  5747
                        5753#  5756   5783#  5785#  5801#  5802   5823#  5824   5831#  5834#  5837#  5838   5845#
                        5846   5872#  5874#  5894#  5895   5902#  5905#  5908#  5909   5916#  5917   5923#  5924
                        5932#  5933   5951#  5953#  5972#  5973   5980#  5983#  5986#  5987   5994#  5995   6001#
                        6002   6010#  6011   6036#  6038#  6054#  6055   6076#  6077   6084#  6087#  6090#  6091
                        6098#  6099   6125#  6127#  6147#  6148   6155#  6158#  6161#  6162   6169#  6170   6176#
                        6177   6185#  6186   6204#  6206#  6225#  6226   6233#  6236#  6239#  6240   6247#  6248
                        6254#  6255   6263#  6264   6293#  6295#  6311#  6312   6340#  6341   6348#  6351#  6354#
                        6355   6362#  6363   6382#  6384#  6406#  6407   6414#  6417#  6420#  6421   6428#  6429
                        6449#  6450   6468#  6469   6485#  6486   6497#  6498   6515#  6517#  6538#  6539   6546#
                        6549#  6552#  6553   6560#  6561   6580#  6581   6599#  6600   6616#  6617   6628#  6629
                        6657#  6659#  6702#  6704#  6720#  6721   6749#  6750   6757#  6760#  6763#  6764   6771#
                        6772   6791#  6793#  6815#  6816   6823#  6826#  6829#  6830   6837#  6838   6857#  6858
                        6866#  6867   6884#  6886#  6906#  6907   6914#  6917#  6920#  6921   6928#  6929   6948#
                        6949   6957#  6958   6964#  6967   6993#  6995#  7038#  7053#  7055#  7095#  7096   7103#
                        7106#  7109#  7110   7117#  7118   7126#  7127   7140#  7141   7159#  7161#  7200#  7201
                        7208#  7211#  7214#  7215   7222#  7223   7231#  7232   7245#  7246   7277#  7291#  7293#
                        7325#  7326#  7327   7332#  7333   7346#  7349#  7350   7357#  7358   7365#  7366   7384#
                        7386#  7416#  7418#  7419   7426#  7427   7441#  7444#  7445   7452#  7453   7459#  7460
                        7485#  7487#  7503#  7504   7525#  7526   7533#  7536#  7539#  7540   7547#  7548   7567#
                        7569#  7596#  7597   7604#  7607#  7610#  7611   7618#  7619   7625#  7626   7634#  7635
                        7653#  7655#  7678#  7679   7686#  7689#  7692#  7693   7700#  7701   7707#  7708   7716#
                        7717   7723#  7726   7750#  7752#  7768#  7769   7790#  7791   7798#  7801#  7804#  7805
                        7812#  7813   7832#  7834#  7856#  7857   7864#  7867#  7870#  7871   7878#  7879   7885#
```

```
                        7886  7894#  7895  7913#  7915#  7936#  7937  7944#  7947#  7950#  7951  7958#  7959
                        7965#  7966  7974#  7975  7981#  7984  8014#  8016#  8032#  8033  8054#  8055  8062#
                        8065#  8068#  8069  8076#  8077  8096#  8098#  8120#  8121  8128#  8131#  8134#  8135
                        8142#  8143  8149#  8150  8158#  8159  8177#  8179#  8198#  8199  8206#  8209#  8212#
                        8213  8220#  8221  8227#  8228  8236#  8237  8262#  8264#  8280#  8281  8302#  8303
                        8310#  8313#  8316#  8317  8324#  8325  8344#  8346#  8368#  8369  8376#  8379#  8382#
                        8383  8390#  8391  8397#  8398  8406#  8407  8425#  8427#  8447#  8448  8455#  8458#
                        8461#  8462  8469#  8470  8476#  8477  8485#  8486  8492#  8495  8519#  8521#  8537#
                        8538  8559#  8560  8567#  8570#  8573#  8574  8581#  8582  8601#  8603#  8625#  8626
                        8633#  8636#  8639#  8640  8647#  8648  8654#  8655  8663#  8664  8682#  8684#  8704#
                        8705  8712#  8715#  8718#  8719  8726#  8727  8733#  8734  8742#  8743  8749#  8752
                        8776#  8778#  8794#  8795  8816#  8817  8824#  8827#  8830#  8831  8838#  8839  8858#
                        8860#  8882#  8883  8890#  8893#  8896#  8897  8904#  8905  8911#  8912  8920#  8921
                        8939#  8941#  8961#  8962  8969#  8972#  8975#  8976  8983#  8984  8990#  8991  8999#
                        9000  9006#  9009
FMT16 = 010000          1067#  3633  3890  4138  4430  4687  4935  5227  5480  5796  6049  6306  6439
                        6570  6653  6715  6803  6990  7034  7273  7498  7763  8027  8275  8532  8789
                        9491  9920  9922  9993  11250  12326
FNCDTB  070674         10604  13837#
FNCMSK= 000077           910#  12037  12481
F0    = 000002           908#  10503
F1    = 000004           907#  10503
F2    = 000010           906#  10503  11077
F3    = 000020           905#  10503
F4    = 000040           904#  10503
GENBUF  042640          3646  3903  4151  4443  4700  4948  5240  5293  5493  5809  5862  6062  6115
                        6319  6655  6728  6992  7511  7776  8040  8288  8545  8802  9904#
GET     043536          3585  3674  3740  3819  3931  3997  4075  4179  4245  4326  4361  4471  4537
                        4616  4728  4794  4872  4976  5042  5123  5159  5268  5339  5417  5521  5573
                        5614  5681  5722  5837  5908  5986  6090  6161  6239  6354  6420  6552  6763
                        6829  6920  7109  7214  7332  7349  7426  7444  7539  7610  7692  7804  7870
                        7950  8068  8134  8212  8316  8382  8461  8573  8639  8718  8830  8896  8975
                        9314  9338  9365  9388  9414  9518  9551  9587  9621  10121#
GETBUF  001330          1379#  10133
GETINX  001506          1450#  3582*  3583*  4359*  4360*  5156*  5157*  9336*  9337*  9386*  9387*  10079  10134
GETSTS  043452          3668  3734  3813  3925  3991  4069  4173  4239  4320  4465  4531  4610  4722
                        4788  4866  4970  5036  5117  5262  5333  5411  5515  5571  5679  5831  5902
                        5980  6084  6155  6233  6348  6414  6546  6757  6823  6914  7038  7103  7208
                        7277  7325  7416  7533  7604  7686  7798  7864  7944  8062  8128  8206  8310
                        8376  8455  8567  8633  8712  8824  8890  8969  9313  9364  9407  9505  10075#
GNS   = ****** U        1217  3250  9055  9062  13721  13722  13723  13724  13725  13726  13728  13730  13731
                       13732  13733  13734  13735
GO    = 000001           909#  3634  3716  3795  3891  3973  4051  4139  4221  4302  4431  4513  4592
                        4688  4770  4848  4936  5018  5099  5228  5315  5393  5481  5562  5593  5670
                        5699  5797  5884  5962  6050  6137  6215  6307  6396  6528  6664  6716  6804
                        6895  7000  7084  7189  7274  7313  7405  7499  7585  7668  7764  7846  7926
                        8028  8110  8188  8276  8358  8436  8533  8615  8694  8790  8872  8951  9348
                        9397  9509  9542  9576  9610  10291  10402  10414  10419  10435  10525  10528  12482
GTSWR = 104407          3245  3466  13728#
HCE   = 000200           989#  998  6446  6457  6577  6588  6844  6847  6935  6938  9655  10853  10858
                       10873  10875  10888  10903  12488  12518  12521  12531  12534  13837
HCI   = 002000          1069#  6803  10851  12484
HCRC  = 000400           988#  998  9655  10860  12488  12492  12495  12531  12534
HELP    107546          3327  13837#
HELPQS  067605          3321  13837#
HT    = 000011           796#  9159  13198  13239
```

```
IAE    = 002000            986# 10757  11054  11069  11095  11241  11258  11757  11811  11814  12315  12330  13837
IDXMSK= 000077            1145# 10147  10150  10220  10223
IE     = 000100            1159#  1192#
ILF    = 000001             996# 10629  10631  11651  11711  11715  11757  11778  11781  12271  12288  12291  13837
ILF02  = 000002             915#
ILF24  = 000024             925#
ILF26  = 000026             926#
ILF30  = 000030             928#
ILF32  = 000032             929#
ILF34  = 000034             930#
ILF36  = 000036             931#
ILF40  = 000040             932#
ILF42  = 000042             933#
ILF44  = 000044             934#
ILF46  = 000046             935#
ILF54  = 000054             938#
ILF56  = 000056             939#
ILF64  = 000064             942#
ILF66  = 000066             943#
ILF74  = 000074             946#
ILF76  = 000076             947#
ILR    = 000002             995# 11651  11697  11701  11965  11969  11972  12271  12274  12277
ILRG50= 000050            1131#
ILRG52= 000052            1132#
ILRG54= 000054            1133#
ILRG56= 000056            1134#
ILRG60= 000060            1135#
ILRG62= 000062            1136#
ILRG64= 000064            1137#
ILRG66= 000066            1138#
ILRG70= 000070            1139#
ILRG72= 000072            1140#
ILRG74= 000074            1141#
ILRG76= 000076            1142#
IOTVEC= 000020             891#  3196*  3197*
IPCK0  = 000001           1196#
IPCK1  = 000002           1195#
IPCK2  = 000004           1194#
IPCK3  = 000010           1193#
IR     = 000100           1173# 11514
IVC    = 010000           1102#  7124   7136   7229   7239  10672  10673  11276  11399  11825  11832  11835  11904
                         12254  12258  12686  12767  13837
LBC    = 002000           1104# 12372  12390  12395
LBT    = 002000            972# 10832  11056  11058
LF     = 000012            797#  9157  12942  13233  13239  13837
LS     = 000004           1051# 11538  12090
LSC    = 004000           1103# 12372  12406  12409
LST    = 000002           1052# 11538  12090
MCLK   = 004000           1024#
MCPE   = 020000           1153# 10402  10419  10432  10435  12169  12172
MDF    = 000100           1029#
MDPE   = 000400           1170# 10928  12578  12581
MEDENB  001474            1438#  3478*  9347*  9470  9726*
MFGFIL  105526            9492   9695   9708*  9715*  9776  13837#
MI     = 000004           1032#
MIXED   071004            8543   8800  13837#
```

```
MOC    = 000400          1027#
MOM    = 020000          1059#
MOL    = 010000           970#  9647  11337  11347  11348  11353  11358  11386  11637  11640  11801  11878  11879
                         11886  11891  12050  12231  12392  12645  12646  12668  12673  12744  12745  12749  12754
MRD    = 002000          1025#
MS     = 000040          1030#
MSC    = 000002          1033#
MSE    = 100000          1110#  6441   6572   9695   9918
MSER   = 000200          1028#
MUR    = 001000          1026#
MWD    = 000010          1050#  11539  12091
MWP    = 000010          1031#
MXF    = 001000          1169#  10817  12192  12206  12209
NDTMSK= 115760           998#
NED    = 010000          1166#  3277   3547   10384  10388  11166
NEM    = 004000          1167#  10802
NOP    = 000000           914#
NOTEX    070650          3285   13837#
NSA    = 100000          1057#
OCC    = 100000          1038#
OFD    = 000200          1070#
OFFSET= 000014           920#
OM     = 000001           977#  11878  11934  11937  12049  12593  12596
ONES     071044         4441   4698   4946   5291   5491   5807   6113   6726   7509   8038   13837#
OPE    = 020000          1101#
OPI    = 020000           983#  10656  11331  11355  11651  11669  11673  11757  11794  11797  11888  12226  12229
                         12670  12751  13837
OR     = 000200          1172#
PACACK= 000022           924#  9542
PAKACK= 000022           923#   924   9348
PAR    = 000010           993#  9658   10447  10451  10463  11223  11757  11762  11767  12183  12188
PAT    = 000020          1175#
PDA    = 000400          1045#
PGE    = 002000          1168#
PGM    = 001000           973#  12049
PHA    = 000200          1046#
PIP    = 020000           969#  7337   7340   7431   7434   9395   9573   11347  11368  11373  11878  11949  11954
                         12049  12645  12651  12656  12744  12781  12786
PIRQ   = 177772           803#
PIRQVE= 000240           897#
PLFS   = 002000          1043#
PRIERR   044532          3748   3827   4005   4083   4253   4334   4545   4624   4802   4880   5050   5131   5347
                         5425   5622   5730   5916   5994   6169   6247   6428   6560   6837   6928   7117   7222
                         7357   7452   7618   7700   7878   7958   8142   8220   8390   8469   8647   8726   8904
                         8983   10350#
PROMPT   067577          3402   3411   13837#
PR0    = 000000           820#
PR1    = 000040           821#
PR2    = 000100           822#
PR3    = 000140           823#  12917
PR4    = 000200           824#
PR5    = 000240           825#
PR6    = 000300           826#  3183   3473   3495   3541   10136  10209  10249  10281  11158  11442  12923
PR7    = 000340           827#
PS     = 177776           800#   801
PSEL   = 002000          1155#
```

```
PSW    = 177776            801#
PUT      044006           3660   3726   3805   3917   3983   4061   4165   4231   4312   4457   4523   4602   4714
                          4780   4858   4962   5028   5109   5254   5325   5403   5507   5563   5603   5671   5710
                          5823   5894   5972   6076   6147   6225   6340   6406   6538   6673   6749   6815   6906
                          7009   7065   7074   7095   7171   7180   7200   7303   7326   7395   7418   7525   7596
                          7678   7790   7856   7936   8054   8120   8198   8302   8368   8447   8559   8625   8704
                          8816   8882   8961   9351   9400   9510   9543   9577   9611  10197#
PUTBUF   001400           1406#  9481   9730*  10206
PUTINX   001535           1455#  3649   3717   3796   3906   3974   4052   4154   4222   4303   4446   4514   4593
                          4703   4771   4849   4951   5019   5100   5243   5316   5394   5496   5594   5701   5812
                          5885   5963   6065   6138   6216   6329   6397   6529   6665   6738   6806   6897   7001
                          7062*  7063*  7071*  7072*  7086   7168*  7169*  7177*  7178*  7191   7300*  7301*  7314
                          7392*  7393*  7406   7514   7586   7669   7779   7847   7927   8043   8111   8189   8291
                          8359   8438   8548   8616   8695   8805   8873   8952   9349*  9350*  9398*  9399*  9494
                         10207  10454
PWRVEC=  000024            892#  3202*  3203*  13740*  13741*  13750*  13756*  13768*  13769*
QSTMRK   067603           3316   3421  13837#
RCLSTS   054472           9421   9596  11748#
RD     = 000070            944#  3795   4592   5699   6528   6895   7189   7405   7668   7926   8436   8694   8951
RDCHR  = 104411           3308   3322   3332   3335   3403   3412  13593  13731#
RDLIN  = 104412          13663  13732#
RDOCT  = 104413           3346   3362   3383  13733#
RDY    = 000200           1158#  10291  10292  10398  10401  11489
READY    007100           3313   3470#  9030   12935
RECAL  = 000006            917#  9397   9576
RESREG=  104415           9226  13735#
RESVEC=  000010            887#
REX    = 010000           1041#
RG     = 040000           1039#
RGDTPT   071004          13837#
RH     = 000072            945#  9610
RIP    = 000020            922#
RLEASE=  000012            919#
RMAS   = 000016           1119#
RMASI    001346           1389#  12070  13837
RMASO    001416           1416#
RMBA   = 000004           1201#  3500*  3501   3652   3721   3800   3909   3978   4056   4157   4226   4307   4449
                          4518   4597   4706   4775   4853   4954   5023   5104   5246   5320   5398   5499   5598
                          5705   5815   5889   5967   6068   6142   6220   6332   6401   6533   6669   6741   6810
                          6901   7005   7090   7195   7318   7410   7517   7591   7673   7782   7851   7931   8046
                          8115   8193   8294   8363   8442   8551   8620   8699   8808   8877   8956   9499
RMBAE  = 000050           1204#
RMBAI    001334           1384#  4372   5169   11004  11006  11500  13837
RMBAO    001404           1411#  3631*  3714*  3794*  3888*  3971*  4136*  4220*  4428*  4511*  4591*  4685*  4768*
                          4933*  5017*  5225*  5289*  5478*  5592*  5700*  5794*  5858*  6047*  6112*  6304*  6394*
                          6526*  6651*  6713*  6805*  6896*  6988*  7032*  7085*  7190*  7271*  7496*  7584*  7667*
                          7761*  7845*  7924*  8025*  8109*  8273*  8357*  8437*  8530*  8614*  8693*  8787*  8870*
                          8950*  9492*  9910   9997   11000  11003
RMCC   = 000036           1126#
RMCCI    001366           1397#
RMCCO    001436           1424#
RMCS1  = 000000           1115#  1199#  3279   3497*  3543   3655   3723   3802   3912   3980   4058   4160   4228
                          4309   4452   4520   4599   4709   4777   4855   4957   5025   5106   5249   5322   5400
                          5502   5600   5707   5818   5891   5969   6071   6144   6222   6335   6403   6535   6671
                          6744   6812   6903   7007   7092   7197   7320   7412   7520   7593   7675   7785   7853
                          7933   8049   8117   8195   8297   8365   8444   8554   8622   8701   8811   8879   8958
```

		9349	9398	9500	10138	10211	10290	11104						
RMCS11	001330	1382#	10377	10379	10381	10398	10400	10403	10414	10418	10420	10432	10434	10436
		10525	10527	10538	10581	10987	11487	12036	12169	12171	12173	12192	13837	
RMCS10	001400	1409#	3634*	3716*	3795*	3891*	3973*	4051*	4139*	4221*	4302*	4431*	4513*	4592*
		4688*	4770*	4848*	4936*	5018*	5099*	5228*	5290*	5315*	5393*	5481*	5562*	5593*
		5670*	5699*	5797*	5861*	5884*	5962*	6050*	6110*	6137*	6215*	6307*	6396*	6528*
		6654*	6664*	6716*	6804*	6895*	6991*	7000*	7035*	7084*	7189*	7274*	7313*	7405*
		7499*	7585*	7668*	7764*	7846*	7926*	8028*	8110*	8188*	8276*	8358*	8436*	8533*
		8615*	8694*	8790*	8872*	8951*	9348*	9397*	9509*	9542*	9576*	9610*	9756	9914
		10502	11018	12460	12480									
RMCS2 =	000010	1202#	3274*	3275*	3276	3502*	3503	3544	7587	10137	10210	11163*	11165	11444*
		11446*												
RMCS21	001340	1386#	4341	4350	4352	5138	5147	5149	10359	10384	10386	10387	10573	10644
		10745	10788	10803	10818	10929	11001	11509	12206	12208	12210	12578	12580	12582
		12632	12634	12636	13837									
RMCS20	001410	1413#	7582*	7666*										
RMCS3 =	000052	1205#												
RMDA =	000006	1116#	3651	3718	3797	3908	3975	4053	4156	4223	4304	4448	4515	4594
		4705	4772	4850	4953	5020	5101	5245	5317	5395	5498	5595	5702	5814
		5886	5964	6067	6139	6217	6331	6398	6530	6666	6740	6807	6898	7002
		7087	7192	7315	7407	7516	7588	7670	7781	7848	7928	8045	8112	8190
		8293	8360	8439	8550	8617	8696	8807	8874	8953	9495			
RMDA1	001336	1385#	11100	13837										
RMDA0	001406	1412#	3630*	3887*	4135*	4427*	4684*	4932*	5224*	5477*	5793*	6046*	6303*	6650*
		6712*	6987*	7031*	7270*	7495*	7760*	8024*	8272*	8529*	8786*	9014*	9015	9488*
		9669*	9670	9672	9690	9698*	9750	9877*	9913	11026	11027	11246	11248	11252
		12320	12322	12324										
RMDB =	000022	1203#	3504*	3505	4359	5156	10086							
RMDB1	001352	1391#	4368	5158*	5165									
RMDB0	001422	1418#												
RMDC =	000034	1125#	3650	3719	3799	3907	3976	4055	4155	4224	4306	4447	4516	4596
		4704	4773	4852	4952	5021	5103	5244	5318	5397	5497	5596	5704	5813
		5887	5966	6066	6140	6219	6330	6399	6532	6668	6739	6808	6900	7004
		7088	7194	7316	7409	7515	7589	7672	7780	7849	7930	8044	8113	8192
		8292	8361	8441	8549	8618	8698	8806	8875	8955	9496			
RMDC1	001364	1396#	11112	13837										
RMDC0	001434	1423#	3629*	3886*	4134*	4426*	4683*	4931*	5223*	5476*	5540	5544*	5560	5561*
		5585*	5649	5652*	5668	5669*	5693*	5792*	6045*	6302*	6649*	6711*	6986*	7030*
		7269*	7494*	7759*	7989*	7990	8023*	8271*	8528*	8785*	9489*	9749	9876*	9912
		11028	11243	12317										
RMDS =	000012	1117#	9336	9386										
RMDS1	001342	1387#	7337	7339	7341	7431	7433	7435	9345	9395	9540	9573	9647	10416
		10512	10514	10559	10576	10610	10614	10670	10697	10832	11057	11282	11337	11346
		11353	11357	11359	11368	11372	11374	11383	11385	11387	11397	11401	11403	11582
		11623	11625	11627	11637	11639	11641	11801	11839	11877	11886	11890	11892	11902
		11906	11907	11918	11920	11922	11934	11936	11938	11949	11953	11955	12048	12231
		12233	12262	12392	12458	12593	12595	12597	12644	12651	12655	12657	12668	12672
		12674	12684	12688	12690	12743	12749	12753	12755	12765	12769	12770*	12771	12781
		12785	12787	13837										
RMDS0	001412	1414#												
RMDT =	000026	1122#	3582											
RMDT1	001356	1393#	3584*	3589	3591	3593	3595	13837						
RMDT0	001426	1420#												
RMEC1 =	000044	1129#												
RMEC11	001374	1400#	9998	13837										
RMEC10	001444	1427#												

```
RMEC2 = 000046          1130#  10081
RMEC21  001376          1401#  10019  10080  11550  12114  13837
RMEC20  001446          1428#
RMER1 = 000014          1118#   7300   7392  10455
RMER11  001344          1388#   6446   6456   6465   6476   6577   6587   6596   6607   6844   6846   6935   6937
                         9654   9987   9989  10447  10462  10464  10554  10630  10657  10700  10715  10758  10835
                        10861  10876  10891  10944  10963  10966  11054  11069  11084  11095  11223  11230  11257
                        11331  11334  11355  11527  11651  11655  11657  11658  11669  11671  11672  11683  11685
                        11686  11697  11699  11700  11711  11713  11714  11757  11762  11766  11768  11778  11780
                        11782  11794  11796  11798  11811  11813  11815  11888  11965  11969  11971  11973  11983
                        11985  11987  11997  12001  12003  12061  12183  12187  12189  12226  12228  12230  12271
                        12274  12276  12278  12288  12290  12292  12302  12304  12306  12329  12419  12422  12426
                        12428  12437  12439  12441  12451  12453  12454  12488  12492  12494  12496  12505  12507
                        12509  12518  12520  12522  12531  12533  12535  12553  12555  12557  12563  12619  12621
                        12623  12670  12751  12862  13837
RMER10  001414          1415#   7302*  7394*  10451
RMER2 = 000042          1128#
RMER21  001372          1399#   6482   6492   6613   6623   6849   6851   6940   6942   7124   7134   7135   7229
                         7238   7240   9656   9661  10449  10556  10674  10730  10906  11225  11276  11279  11290
                        11318  11321  11370  11399  11572  11764  11825  11832  11834  11836  11849  11851  11853
                        11863  11865  11867  11904  11951  11999  12123  12185  12254  12257  12259  12343  12347
                        12372  12376  12378  12380  12390  12394  12396  12406  12408  12410  12424  12606  12608
                        12610  12653  12686  12767  12783  12797  12803  12805  12807  12817  12819  12821  12865
                        13837
RMER20  001442          1426#
RMLA  = 000020          1120#
RMLA1   001350          1390#  13837
RMLA0   001420          1417#
RMMR1 = 000024          1121#   7062   7071   7168   7177
RMMR11  001354          1392#  11537  12089  13837
RMMR10  001424          1419#   7064*  7073*  7170*  7179*
RMMR2 = 000040          1127#
RMMR21  001370          1398#  11559  12101  13837
RMMR20  001440          1425#
RMOF  = 000032          1124#   3654   3720   3798   3911   3977   4054   4159   4225   4305   4451   4517   4595
                         4708   4774   4851   4956   5022   5102   5248   5319   5396   5501   5597   5703   5817
                         5888   5965   6070   6141   6218   6334   6400   6531   6667   6743   6809   6899   7003
                         7089   7193   7317   7408   7519   7590   7671   7784   7850   7929   8048   8114   8191
                         8296   8362   8440   8553   8619   8697   8810   8876   8954   9498
RMOF1   001362          1395#   9991   9993  10851  10961  12484  12560  13837
RMOF0   001432          1422#   3633*  3890*  4138*  4430*  4687*  4935*  5227*  5480*  5796*  6049*  6306*  6653*
                         6715*  6803*  6990*  7034*  7273*  7498*  7763*  8027*  8275*  8532*  8789*  9491*  9920
                        11250  12326  13837
RMR   = 000004           994#  11651  11683  11687  11965  11983  11986  12271  12302  12305
RMSN  = 000030          1123#
RMSN1   001360          1394#  13837
RMSN0   001430          1421#
RMWC  = 000002          1200#   3498*  3499   3653   3722   3801   3910   3979   4057   4158   4227   4308   4450
                         4519   4598   4707   4776   4854   4955   5024   5105   5247   5321   5399   5500   5599
                         5706   5816   5890   5968   6069   6143   6221   6333   6402   6534   6670   6742   6811
                         6902   7006   7091   7196   7319   7411   7518   7592   7674   7783   7852   7932   8047
                         8116   8194   8295   8364   8443   8552   8621   8700   8809   8878   8957   9497
RMWC1   001332          1383#  10036  10985  10997  11015  11079  13837
RMWC0   001402          1410#   3632*  3715*  3889*  3972*  4137*  4219*  4429*  4512*  4686*  4769*  4934*  5016*
                         5226*  5288*  5479*  5591*  5795*  5857*  6048*  6111*  6305*  6395*  6527*  6652*  6714*
                         6989*  7033*  7272*  7497*  7583*  7762*  7844*  7925*  8026*  8108*  8274*  8356*  8531*
```

```
                    8613*  8788*  8871*  9490*  9752   9911   10037  10998  11016
RQA    = 100000     1079#  11560  12102
RQB    = 040000     1080#  11560  12102
RTC    = 000016      921#
SADMSK= 000037       963#
SAVREG= 104414      9097   13734#
SA1    = 000001      958#
SA16   = 000020      954#
SA2    = 000002      957#
SA4    = 000004      956#
SA8    = 000010      955#
SC     = 100000     1151#  10402  10419  10435  11488
SCTMSG  067464      3640    3897   4145   4437   4694   4942   5234   5487   5803   6056   6313   6722   7505
                    7770    8034   8282   8539   8796  13837#
SCTMSK= 003700      1014#
SC0    = 000100     1012#
SC1    = 000200     1011#
SC2    = 000400     1010#
SC3    = 001000     1009#
SC4    = 002000     1008#
SEARCH= 000030       927#  10578  10979
SECERR  045364      3764    3843   4021   4099   4269   4384   4561   4640   4818   4896   5066   5181   5363
                    5441    5638   5746   5932   6010   6185   6263   6497   6628   6866   6957   7140   7245
                    7365    7459   7634   7716   7894   7974   8158   8236   8406   8485   8663   8742   8920
                    8999   10498#
SEEK   = 000004       916#  5562   5670
SEKSTS  051440      5579    5687  11212#
SHUT    063412      9078   12932#
SHUT2   063404     12931#  13434  13517
SIZCLK  044224      3453   10245#
SKI    = 040000     1100#  11242  11245  11261  11291  11293  11297  11308  11370  11573  11825  11849  11852
                   11951   12316  12319  12333  12344  12346  12347  12349  12361  12653  12783  12798  12817
                   12820
SNGPRT= 020024      1062#   3589   3593   3597
STACK  = 001100      791#   3194   3487   3527   3571   3609   3866   4114   4406   4663   4911   5203   5456
                    5772    6025   6278   6689   7024   7262   7474   7739   8003   8251   8508   8765
STANDA  006120      3263   3298#
START   005324      1237   3178#  3516  12941
STCDRV  062436     12736#
STKLMT= 177774       802#
STOP    063354     12914#  13255
STSD1   105416     13837#
STSD2   105434     13837#
STSD3   105454     13837#
STSD4   105466     13837#
STSF    105517     13837#
STSH1   105061     13837#
STSH2   105136     13837#
STSH3   105225     13837#
STSH4   105263     13837#
SWR     001154      1290#   3192   3214*  3216   3222*  3229*  3243   3464   9098  13256  13270  13273  13288
                   13295   13357  13364  13376  13380  13437  13476  13531*  13748  13761*
SWREG   000176      1219#   3222   3243   3464  13437  13476  13499
SW0    = 000001      855#
SW00   = 000001      845#    855
SW01   = 000002      844#    854
```

```
SW02   = 000004            843#    853
SW03   = 000010            842#    852
SW04   = 000020            841#    851
SW05   = 000040            840#    850
SW06   = 000100            839#    849
SW07   = 000200            838#    848
SW08   = 000400            837#    847
SW09   = 001000            836#    846
SW1    = 000002            854#
SW10   = 002000            835#
SW11   = 004000            834#
SW12   = 010000            833#
SW13   = 020000            832#   9098
SW14   = 040000            831#
SW15   = 100000            830#
SW2    = 000004            853#
SW3    = 000010            852#
SW4    = 000020            851#
SW5    = 000040            850#
SW6    = 000100            849#
SW7    = 000200            848#
SW8    = 000400            847#
SW9    = 001000            846#
TADMSK= 003400             962#
TAG    = 020000           1081#
TAGADR= 001114            1262#   1269
TAP    = 040000           1058#
TA1    = 000400            953#
TA2    = 001000            952#
TA4    = 002000            951#
TBITVE= 000014             888#
TIMOUT  044346            3671    3737    3816    3928    3994    4072    4176    4242    4323    4468    4534    4613    4725
                          4791    4869    4973    5039    5120    5265    5336    5414    5518    5572    5611    5680    5719
                          5834    5905    5983    6087    6158    6236    6351    6417    6549    6679    6760    6826    6917
                          7015    7106    7211    7346    7441    7536    7607    7689    7801    7867    7947    8065    8131
                          8209    8313    8379    8458    8570    8636    8715    8827    8893    8972    9408    9586    9620
                         10274#
TKVEC  = 000060            895#   13414*  13415*
TPVEC  = 000064            896#
TRAPVE= 000034             894#    3200*   3201*
TRE    = 040000           1152#   10402   10419   10435   10580   10582   10987
TRTVEC= 000014             889#
TST    = 010000           1082#   11561   12103
TSTNMB  037752             9109*   9110*   9112    9229#
TSTPRP  040020             3620    3702    3783    3877    3959    4040    4125    4207    4291    4417    4499    4580    4674
                           4756    4837    4922    5004    5088    5214    5303    5382    5467    5552    5660    5783    5872
                           5951    6036    6125    6204    6293    6382    6515    6657    6702    6791    6884    6993    7053
                           7159    7291    7384    7485    7567    7653    7750    7832    7913    8014    8096    8177    8262
                           8344    8425    8519    8601    8682    8776    8858    8939    9277#
TSTQUE  001450            1434#   3436    3437*   3477    3489    3529    3573    3611    3868    4116    4408    4665    4913
                           5205    5458    5774    6027    6280    6691    7026    7264    7476    7741    8005    8253    8510
                           8767    9025    9027*   9031*  11160   11445   11512   12073
TST1    007136             3482#  13315
TST10   014200             4660#  13322
TST11   015172             4908#  13323
TST12   016434             5200#  13324
```

```
TST13   017454          5453#  13325
TST14   021002          5769#  13326
TST15   022022          6022#  13327
TST16   023042          6275#  13328
TST17   024740          6686#  13329
TST2    007334          3524#  13316
TST20   026414          7021#  13330
TST21   027522          7259#  13331
TST22   030470          7471#  13332
TST23   031546          7736#  13333
TST24   032634          8000#  13334
TST25   033626          8248#  13335
TST26   034650          8505#  13336
TST27   035672          8762#  13337
TST3    007520          3568#  13317
TST4    007702          3606#  13318
TST5    010724          3863#  13319
TST6    011716          4111#  13320
TST7    013156          4403#  13321
TYPBN = 104406          9219   13726#
TYPDS = 104405          9059    9066   9213   13725#
TYPE  = 104401          3237    3257   3285   3307   3310   3316   3321   3324   3327   3331   3334   3342   3345
                        3351    3356   3361   3367   3377   3382   3388   3401   3402   3405   3411   3414   3421
                        3640    3897   4145   4437   4694   4942   5234   5487   5803   6056   6313   6722   7505
                        7770    8034   8282   8539   8796   9053   9060   9067   9102   9103   9111   9120   9126
                        9135    9173   9179   9183   9196   9202   9204   9222   9224   12936  13008  13074  13143
                       13203   13359  13367  13431  13443  13497  13498  13501  13512  13522  13533  13552  13600
                       13606   13611  13615  13620  13621  13623  13626  13630  13721#  13770
TYPOC = 104402          3344    9216   13500  13722#
TYPON = 104404         13724#
TYPOS = 104403          3287    3358   3379   9106   9114   9123   9129   13723#
UBUSQS  067773          3331   13837#
UNS   = 040000           982#   7302   7394   11528  11651  11655  11659  11965  11997  12002  12419  12422  12427
UNTMSK= 000007          1183#  10360  10362
UPE   = 020000          1165#  10744
USE   = 040000          1111#   6441   6572   9695   9918
USRFIL  106536          9709*  9716*  9867   9869   13837#
U0    = 000001          1180#   7580   7665
U1    = 000002          1179#   7580   7665
U2    = 000004          1178#   7580   7665
VV    = 000100           976#   9345   9540   10670  11282  11347  11348  11397  11402  11623  11626  11839  11878
                       11879   11902  11908  12049  12233  12262  12645  12646  12684  12689  12744  12745  12765
                       12770
WC    = 000040          1048#  11538  12090
WCD   = 000050           936#   4051   4302   4848   5099   5393   5962   6215   8188
WCE   = 040000          1164#   4341   4351   5138   5148  10643  13837
WCEHI = 010000          1189#
WCELO = 004000          1190#
WCF   = 000040           991#    998  10714  12619  12622
WCH   = 000052           937#
WD    = 000060           940#   3716   3973   4221   4513   4770   5018   5290   5315   5593   5861   5884   6110
                        6137   6396   6804   7035   7084   7313   7585   7846   8110   8358   8615   8872
WH    = 000062           941#   3634   3891   4139   4431   4688   4936   5228   5481   5797   6050   6307   6654
                        6664   6716   6991   7000   7274   7499   7764   8028   8276   8533   8790   12482
WLE   = 004000           985#    998  10695  10699  10712  10727  10742  12419  12451  12455  12463  13837
WRL   = 004000           971#  10697  12458
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
XDPAG   070563            3257  13837#
XSIZ    005776            3267#  3408
ZEROS   071106            3644   3901   4149   5238   5859   6060   6317   7774  13837#
$APTHD  001100            1249   1255#
$ASTAT= ****** U         13813  13828
$ATYC   067242           13784  13786#
$ATY1   067216           13782#
$ATY3   067224           13188  13783#
$ATY4   067234           13372  13785#
$AUTOB  001150            1287#  3247*  3468*  13490  13647
$BASE   001276            1360#  3273   3343   3353*  3488   3528   3572   3610   3867   4115   4407   4664   4912
                          5204   5457   5773   6026   6279   6690   7025   7263   7475   7740   8004   8252   8509
                          8766  10132  10205  10282  11159  11443  13837
$BDADR  001136            1282#  4372*  4373*  4374   5169*  5170*  5171  10047*  13837
$BDDAT  001142            1284#  4352*  4368*  4378   5149*  5165*  5175   6456*  6476*  6492*  6587*  6607*  6623*
                          6846*  6847*  6851*  6852*  6937*  6938*  6942*  6943*  7134*  7240*  7341*  7435*  10045*
                         10359*  10360*  10363  10381*  10387*  10403*  10420*  10436*  10464*  10514*  10515*  10527*  10528*
                         10538*  10539*  10541  10559*  10560*  10561  10581*  10582*  10583  10614*  10615*  10616  10630*
                         10631*  10632  10644*  10645*  10657*  10658*  10674*  10675*  10700*  10701*  10715*  10716*  10730*
                         10731*  10745*  10746*  10758*  10759*  10788*  10789*  10803*  10804*  10818*  10819*  10835*  10836*
                         10861*  10862*  10876*  10877*  10891*  10892*  10906*  10907*  10929*  10930*  10944*  10945*  10966*
                         10967*  10985*  11006*  11057*  11058*  11059  11084*  11085*  11086  11100*  11101  11112*  11113
                         11230*  11257*  11258*  11259  11279*  11290*  11291*  11297  11308  11321*  11334*  11359*  11373*
                         11374*  11387*  11403*  11487*  11488*  11490  11500*  11509*  11517  11527*  11528*  11537*  11538*
                         11540  11550*  11559*  11560*  11562  11572*  11573*  11582*  11583*  11585  11627*  11641*  11657*
                         11671*  11685*  11699*  11713*  11768*  11782*  11798*  11815*  11836*  11853*  11867*  11892*  11907*
                         11922*  11938*  11955*  11973*  11987*  12003*  12036*  12037*  12039  12048*  12049*  12051  12061*
                         12070*  12077*  12080  12089*  12090*  12092  12101*  12102*  12104  12114*  12123*  12173*  12189*
                         12210*  12230*  12259*  12278*  12292*  12306*  12329*  12330*  12331  12343*  12344*  12380*  12396*
                         12410*  12428*  12441*  12453*  12496*  12509*  12522*  12535*  12557*  12582*  12597*  12610*  12623*
                         12636*  12657*  12674*  12690*  12755*  12771*  12787*  12807*  12821*  12877*  12892*  13837
$BELL   001212            1307#  13359  13392  13443  13640
$BIN    063640           13004*  13007*  13008  13015#
$CDW1   001302            1362#
$CDW2   001304            1363#
$CHARC  064572           13205*  13215*  13222  13231*  13236#
$CKSWR  065670           13476#  13730
$CMTAG  001114            1270#   3189   3190   3198   3204   3205   3206
$CM3  = 000000            1300#
$CM4  = 000005            1300#   1301#  1302#  1303#  1304#  1305#
$CNTLC  066566           13431  13512  13640#
$CNTLG  066600           13497  13642#
$CNTLU  066573           13522  13615  13641#
$CPUOP  001250            1334#
$CRLF   001217            1309#   9067   9102   9135   9173   9183   9196   9204   9224  13204  13239  13367  13392
                         13533  13620  13640
$DBLK   064056           13040  13074  13082#
$DDW0   001306            1364#
$DDW1   001310            1365#
$DDW2   001312            1366#
$DDW3   001314            1367#
$DDW4   001316            1368#
$DDW5   001320            1369#
$DDW6   001322            1370#
$DDW7   001324            1371#
$DEVCT  001232            1325#
```

```
$DEVM   001300  1361# 3270* 3282* 3400* 3409* 3427* 3428  3435
$DOAGN  037206  9049  9070  9076#
$DTBL   064046 13043 13078#
$ENDAD  037176  1228  3235  9072# 13387
$ENDCT  037042  3204  9051#
$ENULL  037212  9079#
$ENV    001242  1330# 3241  3262  3462  13183 13369 13791 13815
$ENVM   001243  1331# 3227  3267  13185 13190 13793
$EOP    037006  3517  9041# 12939
$EOPCT  037034  3204* 9048# 9052
$EOSP   036752  3560  3600  9023#
$ERFLG  001117  1273# 13244 13284 13286 13292* 13314 13354* 13392
$ERMAX  001131  1279# 3207* 13286 13309* 13314
$ERROR  065172  3198  13352#
$ERRPC  001132  1280# 9127  13361* 13362* 13363 13392
$ERRTB  001564  1473# 9142
$ERTTL  001126  1277# 9064  9068* 13360* 13392
$ESCAP  001210  1306# 3206* 4280* 4397* 5077* 5194* 5541* 5586* 5650* 5694* 13308* 13383 13385
                13392
$ETABL  001242  1329#
$ETEND  001326  1261  1372#
$FATAL  001224  1322# 13819*
$FFLG   067462 13782* 13785* 13813 13822* 13830#
$FILLC  001172  1298# 13208 13239
$FILLS  001171  1297# 13239
$GDADR  001134  1281# 4371* 4374  5168* 5171  10046* 13837
$GDDAT  001140  1283# 4350* 4351* 4369* 4370* 4378  5147* 5148* 5166* 5167* 5175  6457* 6475*
                6493* 6588* 6606* 6624* 6853* 6944* 7135* 7136* 7238* 7239* 7339* 7340* 7433*
                7434* 10044* 10361* 10362* 10363 10379* 10380* 10386* 10388* 10400* 10401* 10402* 10418*
                10419* 10434* 10435* 10462* 10463* 10516* 10529* 10540* 10541 10553* 10558* 10561 10572*
                10580* 10583 10609* 10612* 10613* 10616 10628* 10629* 10632 10639* 10991* 10997* 10998*
                10999* 11000* 11003* 11004 11051* 11056* 11059 11068* 11081* 11086 11097* 11098* 11099*
                11101 11110* 11111* 11113 11229* 11241* 11254* 11259 11278* 11292* 11293* 11320* 11333*
                11357* 11358* 11372* 11385* 11386* 11401* 11402* 11489* 11490 11499* 11515* 11517 11526*
                11539* 11540 11549* 11561* 11562 11571* 11584* 11585 11625* 11626* 11639* 11640* 11658*
                11659* 11672* 11673* 11686* 11687* 11700* 11701* 11714* 11715* 11766* 11767* 11780* 11781*
                11796* 11797* 11813* 11814* 11834* 11835* 11851* 11852* 11865* 11866* 11890* 11891* 11906*
                11908* 11920* 11921* 11936* 11937* 11953* 11954* 11971* 11972* 11985* 11986* 12001* 12002*
                12038* 12039 12050* 12051 12060* 12091* 12092 12103* 12104 12112* 12171* 12172* 12187*
                12188* 12208* 12209* 12228* 12229* 12257* 12258* 12276* 12277* 12290* 12291* 12304* 12305*
                12315* 12328* 12331 12345* 12346* 12378* 12379* 12394* 12395* 12408* 12409* 12426* 12427*
                12439* 12440* 12454* 12455* 12463* 12494* 12495* 12507* 12508* 12520* 12521* 12533* 12534*
                12555* 12556* 12580* 12581* 12595* 12596* 12608* 12609* 12621* 12622* 12634* 12635* 12655*
                12656* 12672* 12673* 12688* 12689* 12753* 12754* 12769* 12785* 12786* 12805* 12806* 12819*
                12820* 12878* 12893* 13837
$GET42  037166  9069#
$GTSWR  065760 13498# 13728
$HD   = 000001   767   768
$HIBTS  001100  1256#
$HIOCT  066730 13680* 13685#
$ICNT   001120  1274# 13299* 13300 13302* 13313
$ILLUP  067200 13740  13756  13773#
$INTAG  001151  1288# 13495* 13514 13534  13647
$ITEMB  001130  1278#  9118  13363* 13371  13392
$LF     001220  1310# 13239  13392 13630  13640
$LFLG   067461 13823* 13829#
```

```
$LPADR  001122          1275#  3208*  3484*  3691*  3773*  3948*  4030*  4196*  4278*  4398*  4488*  4570*  4745*
                        4827*  4993*  5075*  5195*  5285*  5372*  5538*  5647*  5854*  5941*  6107*  6194*  6371*
                        6506*  6681*  6780*  6875*  7016*  7042*  7149*  7254*  7280*  7374*  7556*  7643*  7821*
                        7903*  7995*  8085*  8167*  8333*  8415*  8590*  8672*  8847*  8929*  9020*  13281* 13290*
                       13306*  13311  13313
$LPERR  001124          1276#  3209*  3485*  3692*  3774*  3949*  4031*  4197*  4279*  4399*  4489*  4571*  4746*
                        4828*  4994*  5076*  5196*  5286*  5373*  5539*  5648*  5855*  5942*  6108*  6195*  6372*
                        6507*  6682*  6781*  6876*  7017*  7043*  7150*  7255*  7281*  7375*  7557*  7644*  7822*
                        7904*  7996*  8086*  8168*  8334*  8416*  8591*  8673*  8848*  8930*  9021*  13290  13307*
                       13313   13382
$MADR1  001254          1347#
$MADR2  001260          1351#
$MADR3  001264          1354#
$MADR4  001270          1357#
$MAIL   001222          1257    1261   1320#  3226   3241   3462   3490   3530   3574   3612   3869   4117   4409
                        4666    4914   5206   5459   5775   6028   6281   6692   7027   7265   7477   7742   8006
                        8254    8511   8768  13183  13305  13369
$MAMS1  001252          1341#
$MAMS2  001256          1349#
$MAMS3  001262          1352#
$MAMS4  001266          1355#
$MBADR  001102          1257#
$MFLG   067460         13783*  13789  13824*  13828#
$MNEW   066616         13501   13645#
$MSGAD  001236          1327#  13799*  13802
$MSGLG  001240          1328#  13804*
$MSGTY  001222          1321#  13797  13805*  13817  13821*
$MSWR   066605         13498   13643#
$MTYP1  001253          1342#
$MTYP2  001257          1350#
$MTYP3  001263          1353#
$MTYP4  001267          1356#
$MXCNT  065112         13303   13313#
$NULL   001170          1296#  13210  13239
$NWTST= 000001          3479#   3481   3520#  3522   3564#  3566   3603#  3860#  4108#  4400#  4657#  4905#  5197#
                        5450#   5766#  6019#  6272#  6683#  7018#  7256#  7468#  7733#  7997#  8245#  8502#  8759#
$OCNT   064310         13115*  13144*  13157#
$OMODE  064312         13110*  13114*  13119  13122*  13133*  13159#
$OVER   065076         13257   13282  13291  13301  13310#
$PASS   001230          1324#   3226*  9045*  9046*  9057   9079  13297  13314
$PASTM  001106          1259#
$POWER  067206         13771   13776#
$PWRDN  067040          3202   13740#  13768
$PWRMG  067174         13771#
$PWRUP  067112         13750   13756#
$QUES   001216          1308#  13239  13392  13552  13623  13640
$RDCHR  066232         13565#  13731
$RDDEC= ****** U       13734
$RDLIN  066322         13588#  13732
$RDOCT  066630         13658#  13733
$RDSZ = 000010         13581#
$RESRE  063530         12980#  13735
$RTNAD  037210          9078#
$R2A  = ****** U       13736
$SAVRE  063472         12964#  13734
$SAVR6  067204         13749*  13757  13758*  13759*  13775#
```

```
$SCOPE  064576          3196  13253#
$SETUP= 000137          1210#  3195   3196   3198   3200   3202   3204   3205   3206   3208   3235   3238   3459
                        9043  13254  13353  13379  13387  13435  13440  13441  13471  13647
$STUP = 177777          1210#
$SVLAD  065042         13265  13304#
$SVPC = 000200          1226#  1231
$SWR  = 167400           756#   767    772    773    774    775    776    777    778    779   1305   1306   1307
                        3205   3206   3208   3209   3490   3530   3574   3612   3869   4117   4409   4666   4914
                        5206   5459   5775   6028   6281   6692   7027   7265   7477   7742   8006   8254   8511
                        8768   9038   9044   9071   9077   9079  13245  13246  13247  13248  13249  13256  13268
                       13270  13271  13284  13285  13286  13293  13294  13295  13307  13310  13313  13344  13345
                       13346  13347  13348  13357  13364  13376  13380  13392  13772
$SWREG  001244          1332#  3229
$SWRMK= 000000           779   780  13249  13250  13274
$SWOBT  065114         13280  13314#
$TESTN  001226          1323#  3490*  3530*  3574*  3612*  3869*  4117*  4409*  4666*  4914*  5206*  5459*  5775*
                        6028*  6281*  6692*  7027*  7265*  7477*  7742*  8006*  8254*  8511*  8768*  9110  13305*
$TIMES  001206          1305#  3205*  9044* 13293* 13300  13303* 13313
$TKB    001162          1293#  13395  13416  13427  13452  13480  13507
$TKCNT  065372         13396#  13411* 13441  13458* 13572  13574*
$TKINT  065402          3299   3472  13411#  13432  13493
$TKQEN= 065401         13400#  13466  13577
$TKQIN  065374         13397#  13412* 13413  13464* 13465* 13466  13468*
$TKQOU  065376         13398#  13413* 13575  13576* 13577  13579*
$TKQSR  065400         13399#  13412  13468  13579
$TKS    001160          1292#  13395  13417* 13448* 13450  13456* 13478  13494* 13504  13516* 13536*
$TKSRV  065452         13414  13427#
$TMP0   001174          1300#  3544*  3547   3644*  3901*  4149*  4353*  4441*  4698*  4946*  5150*  5238*  5291*
                        5491*  5807*  5859*  6060*  6113*  6317*  6458*  6589*  6726*  7509*  7774*  8038*  8286*
                        8543*  8800*  9941  10048* 10137* 10210* 11165* 11166  13837
$TMP1   001176          1301#  3309*  3310   3311   3314   3323*  3324   3325   3333*  3334   3337   3347*  3349
                        3353   3354*  3355*  3357   3363*  3365   3369   3370*  3371*  3372*  3373*  3374*  3375*
                        3376*  3378   3384*  3386   3391*  3392*  3393*  3394*  3395*  3396   3404*  3405   3406
                        3413*  3414   3415   3417   3419   3423   3543*  3551   3597*  3645*  3902*  4150*  4354*
                        4442*  4699*  4947*  5151*  5239*  5292*  5492*  5808*  5860*  6061*  6114*  6318*  6459*
                        6590*  6727*  7510*  7775*  8039*  8287*  8544*  8801*  9942  10138*  10139  10211* 10212
                       11164* 11171  12862* 12863* 12875  12877  13837
$TMP2   001200          1302#  3598*  12865* 12866* 12889  12892  13837
$TMP3   001202          1303#
$TMP4   001204          1304#
$TN   = 000030           767#  3479   3490#  3520   3530#  3564   3574#  3603   3612#  3860   3869#  4108   4117#
                        4400   4409#  4657   4666#  4905   4914#  5197   5206#  5450   5459#  5766   5775#  6019
                        6028#  6272   6281#  6683   6692#  7018   7027#  7256   7265#  7468   7477#  7733   7742#
                        7997   8006#  8245   8254#  8502   8511#  8759   8768#  13275  13315
$TPB    001166          1295#  13228* 13239
$TPFLG  001173          1299#  13177  13239
$TPS    001164          1294#  13226  13239
$TRAP   066732          3200   13694#
$TRAP2  066772         13709#  13720
$TRP  = 000016         13713#  13722#  13723#  13724#  13725#  13726#  13727#  13728  13729#  13730  13731#  13732#  13733#
                       13734#  13735#  13736#
$TRPAD  067004         13703   13720#
$TSTM   001104          1258#
$TSTNM  001116          1272#   9043*  13244  13277* 13304* 13305  13310  13314  13356  13392
$TTYIN  066556         13590   13591  13603  13621  13635  13639#
$TYPBN  063566         13001#  13726
```

```
$TYPDS  063642          13028#  13725
$TYPE   064314          13177#  13713   13721   13810
$TYPEC  064526          13207   13214   13221   13226#  13227   13538
$TYPEX  064574          13232   13234   13237#
$TYPOC  064112          13113#  13722
$TYPON  064126          13112   13115#  13724
$TYPOS  064066          13108#  13723
$UNIT   001234           1326#   3477*   9104   10361
$UNITM  001110           1260#
$USWR   001246           1333#
$VECT1  001272           1358#   3354    3369*   3370    3396*
$VECT2  001274           1359#
$XTSTR  064614          13259#
$$GET4= 000000           9071#
$$SW08= 000030          13314#  13315   13316#  13317#  13318#  13319#  13320#  13321#  13322#  13323#  13324#  13325#  13326#
                        13327#  13328#  13329#  13330#  13331#  13332#  13333#  13334#  13335#  13336#  13337#  13338#
$OFILL  064311          13109*  13113*  13123   13158#
$40CAT= ****** U        13256   13366
.       = 111751         1213#   1217#   1226    1227#   1229#   1231#   1236#   1239#   1245    1246#   1248#   1250#   1262
                         1269#   1311    1434#   1450#   1455#   3193    3208    3209    3251#   9056#   9079    9080#  12934
                        12946#  13082#  13239   13313   13314   13392   13395   13399#  13400   13401#  13639#  13640   13647#
                        13752   13774   13831#  13837#
.$ASTA= ****** U        13783   13786
.$X     = 001100         1245#   1250


. ABS.  111751      000


 ERRORS DETECTED:  0

DSKZ:CZRMEC,DSKZ:CZRMEC.SEQ/SOL/CRF/DOC/NL:MC:MD:CND:TOC/LI:ME=DSKZ:CZRMEC.P11
RUN-TIME: 109 104 4 SECONDS
RUN-TIME RATIO: 1138/217=5.2
CORE USED:  34K  (67 PAGES)

DOCUMENT PAGES:  328
```