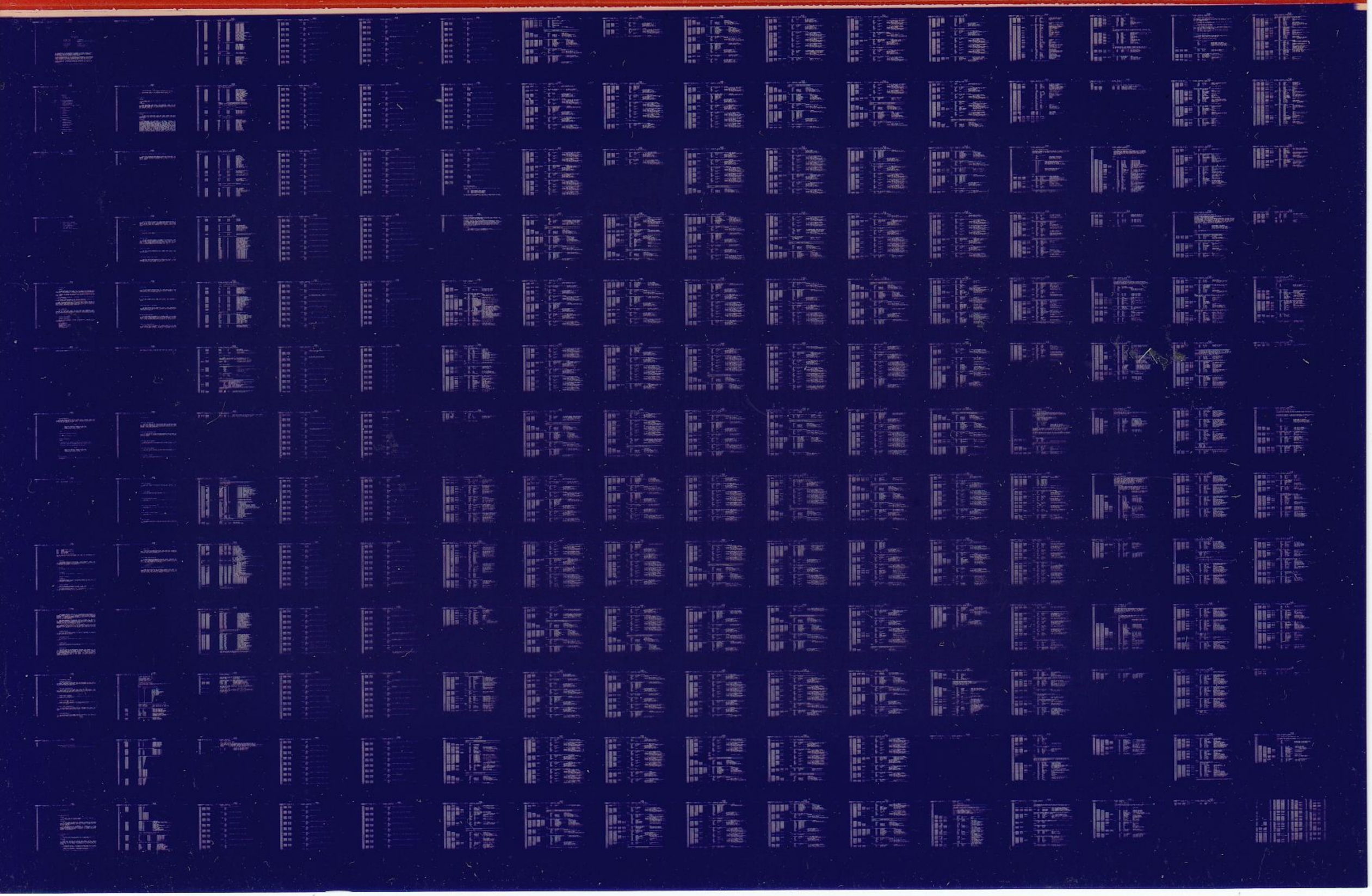
RM03,02

FUNCTIONAL TEST 2
CZRMDB0

AH-B001B-MC
COPYRIGHT © 1977
FICHE 1 OF 2

JAN 1978
digital
MADE IN USA



RM03,02

FUNCTIONAL TEST 2
CZRMDB0

AH-B001B-MC
COPYRIGHT © 1977
FICHE 2 OF 2

JAN 1978
digital
MADE IN USA

```
                                .REM    \
```

IDENTIFICATION

PRODUCT CODE:	AC-B0008-MC
PRODUCT NAME:	CZRMDB0 RM03/RM02 FUNCTIONAL TEST, PART 2
DATE CREATED:	1 AUGUST 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

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

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

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

COPYRIGHT (C) 1977, DIGITAL EQUIPMENT CORPORATION

```
                                .PAGE
```

CONTENTS

109

2. DUAL PORT CONFIGURATIONS

3. MEMORY PARITY HARDWARE

4. MEMORY MANAGEMENT HARDWARE

5. ACT,APT COMPATIBILITY

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

THE RMO3 SUBSYSTEM FUNCTIONAL TEST .3 A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE PMO3 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 RMO3 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 RMO3 DISK SUBSYSTEM WHICH CONSISTS OF THE RHXX MASSBUS CONTROLLER, THE RMO3 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 RMO3 SUBSYSTEM FUNCTIONAL TEST:

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

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

2.2 MEDIA REQUIREMENTS

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

NOTE

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

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, 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.

```
240                             SW15    HALT ON ERROR
241                             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 RMO3 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)??". IF THE UNIBUS ADDRESS OF THE RMO3 IS NON STANDARD, THE OPERATOR SHOULD RESPOND Y, THEN ANSWER SUBSEQUENT QUESTIONS TO SPECIFY THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY. IF THE OPERATOR DOES NOT RESPOND WITH A Y, THE PROGRAM SKIPS TO THE THIRD QUESTION.

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

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

4.3 PROGRESS REPORTS

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

4.4 PERFORMANCE REPORT

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

4.5 PROGRAM HALTS

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

4.6 ERROR REPORTS

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

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

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

5.2 DUAL PORT CONFIGURATIONS

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

5.3 MEMORY PARITY HARDWARE

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

5.4 MEMORY MANAGEMENT HARDWARE

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

5.5 ACT11, APT11 COMPATIBILITY

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

5.6 XXDP COMPATIBILITY

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

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

CONTROLLER ACCESS TEST

PURPOSE:

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

PROCEDURE:

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

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

DEVICE AVAILABLE TEST

PURPOSE:

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

PROCEDURE:

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

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

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

.BUS TIMEOUT - THE MASSBUS CONTROLLER FAILED TO DETECT A

```
467                                     NONEXISTENT DEVICE;
```

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

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

DRIVE TYPE TEST

PURPOSE:

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

PROCEDURE:

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

WRITE/READ HEADER AND DATA (FORMAT) TESTS

PURPOSE:

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

PROCEDURE:

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

```
                                THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.


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



```
                                FORMAT ZEROS - 18
```

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

FORMAT ZEROS - 16

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

ZERO FILL TEST

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

FORMAT CHECK ZEROS

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

FORMAT CHECK ZEROS W/ WCE ERROR

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

FORMAT ONES

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

FORMAT CHECK ONES

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

FORMAT CHECK ONES W/ WCE ERROR

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

FORMAT MULTIPLE SECTORS

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

FORMAT WITH HEAD SWITCHING

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

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

STARTING SECTOR.

FORMAT WITH IMPLIED SEEK

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

FORMAT WITH MIDTRANSFER SEEK

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

FORMAT EACH SECTOR ADDRESS

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

FORMAT EACH TRACK ADDRESS

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

FORMAT PRIME CYLINDERS

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

PAGE 32

FORMAT LAST SECTOR

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

FOR W/ AOE ERROR

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

FORMAT INVALID SECTOR ADDRESS

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

FORMAT INVALID TRACK ADDRESS

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

FORMAT INVALID CYLINDER ADDRESS

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

FORMAT AT OFFSET

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

IVC FORMAT TEST

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

FORMAT ERROR TEST(18 AND 16)

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

FORMAT HCE (FIRST AND SECOND HEADER WORDS)

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

```
   799
```

```
                        ;PROGRAM REVISION #001

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

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

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

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

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

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

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

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

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

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

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

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

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

```
                                ;PACK ACKNOWLEDGE COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;SEARCH COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;WRITE CHECK DATA COMMAND
                                ;WRITE CHECK HEADER AND DATA
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;WRITE DATA COMMAND
                                ;WRITE HEADER AND DATA COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND
                                ;READ DATA COMMAND
                                ;READ HEADER AND DATA COMMAND
                                ;ILLEGAL COMMAND
                                ;ILLEGAL COMMAND

                                ;RMDA   DISK ADDRESS REGISTER

                                ;TRACK ADDRESS 4
                                ;TRACK ADDRESS 2
                                ;TRACK ADDRESS 1
                                ;SECTOR ADDRESS 16
                                ;SECTOR ADDRESS 8
                                ;SECTOR ADDRESS 4
                                ;SECTOR ADDRESS 2
                                ;SECTOR ADDRESS 1

                                ;TRACK,SECTOR MASKS

                                ;TRACK ADDRESS MASK
                                ;SECTOR ADDRESS MASK

                                ;RMDS   DRIVE STATUS REGISTER

                                ;ATTENTION ACTIVE
                                ;COMPOSITE ERROR
                                ;POSITIONING IN PROGRESS
                                ;MEDIUM ON LINE
                                ;WRITE LOCK
                                ;LAST BLOCK TRANSFERRED
                                ;PROGRAMMABLE
                                ;DRIVE PRESENT
                                ;DRIVE READY
                                ;VOLUME VALID
                                ;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
```

```
1080
1081                                    ;       READ ONLY BITS
1082
1083            100000                  OCC     =       BIT15           ;OCCUPIED
1084            040000                  RG      =       BIT14           ;RUN AND GO
1085            020000                  EBL     =       BIT13           ;END OF BLOCK
1086            010000                  REX     =       BIT12           ;EXCEPTION
1087            004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1088            002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1089            001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1090            000400                  PDA     =       BIT08           ;DATA AREA
1091            000200                  PHA     =       BIT07           ;HEADER AREA
1092            000100                  CONT    =       BIT06           ;CONTINUE
1093            000040                  WC      =       BIT05           ;WORD CLOCK
1094            000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1095            000010                  MWD     =       BIT03           ;WRITE DATA BIT
1096            000004                  LS      =       BIT02           ;LAST SECTOR
1097            000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1098            000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1099
1100                                    ;RMDT   DRIVE TYPE REGISTER
1101
1102            100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1103            040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1104            020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1105            004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1106
1107            020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1108            024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1109
1110                                    ;RMOF   OFFSET REGISTER
1111
1112            010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1113            004000                  ECI     =       BIT11           ;ECC INHIBIT
1114            002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1115            000200                  OFD     =       BIT07           ;OFFSET FORWARD
1116
1117
1118                                    ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1119            001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1120
1121                                    ;RMMR2  MAINTENANCE REGISTER #2
1122
1123                                    .       READ ONLY BITS
1124            100000                  RQA     =       BIT15           ;PORT A REQUEST
1125            040000                  RQB     =       BIT14           ;PORT B REQUEST
1126            020000                  TAG     =       BIT13           ;TAG CONTROL
1127            010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1128            004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1129            002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1130            001000                  BB09    =       BIT09           ;TAG BUS
1131            000400                  BB08    =       BIT08           ;TAG BUS
1132            000200                  BB07    =       BIT07           ;TAG BUS
1133            000100                  BB06    =       BIT06           ;TAG BUS
1134            000040                  BB05    =       BIT05           ;TAG BUS
1135            000020                  BB04    =       BIT04           ;TAG BUS
```

```
1136            000010          BB03    =       BIT03           ;TAG BUS
1137            000004          BB02    =       BIT02           ;TAG BUS
1138            000002          BB01    =       BIT01           ;TAG BUS
1139            000001          BB00    =       BIT00           ;TAG BUS
1140
1141
1142                            ;RMER2  ERROR REGISTER 2
1143
1144            100000          BSE     =       BIT15           ;BAD SECTOR ERROR
1145            040000          SKI     =       BIT14           ;SEEK INCOMPLETE
1146            020000          OPE     =       BIT13           ;OPERATOR PLUG ERROR
1147            010000          IVC     =       BIT12           ;INVALID COMMAND ERROR
1148            004000          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
1149            002000          LBC     =       BIT10           ;LOSS OF BIT CLOCK
1150            000200          DVC     =       BIT07           ;DEVICE CHECK
1151            000010          DPE     =       BIT03           ;DATA PARITY ERROR
1152
1153                            .SBTTL  PROGRAM MNEMONICS
1154
1155            100000          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
1156            040000          USE     =       BIT14           ;USER DETECTED SECTOR ERROR
1157
1158                            .SBTTL  RM03 REGISTER INDEX VALUES
1159
1160            000000          RMCS1   =       00              ;CONTROL STATUS REGISTER
1161            000006          RMDA    =       06              ;DISK ADDRESS REGISTER
1162            000012          RMDS    =       12              ;DRIVE STATUS REGISTER
1163            000014          RMER1   =       14              ;ERROR REGISTER 1
1164            00C016          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
1165            000020          RMLA    =       20              ;LOOK AHEAD REGISTER
1166            000024          RMMR1   =       24              ;MAINTENANCE REGISTER
1167            000026          RMDT    =       26              ;DRIVE TYPE REGISTER
1168            000030          RMSN    =       30              ;SERIAL NUMBER REGISTER
1169            000032          RMOF    =       32              ;OFFSET REGISTER
1170            000034          RMDC    =       34              ;DESIRED CYLINDER REGISTER
1171            000036          RMCC    =       36              ;CURRENT CYLINDER REGISTER
1172            000040          RMMR2   =       40              ;MAINTENANCE REGISTER 2
1173            000042          RMER2   =       42              ;ERROR REGISTER 2
1174            000044          RMEC1   =       44              ;ECC POSITION REGISTER
1175            000046          RMEC2   =       46              ;ECC PATTERN REGISTER
1176            000050          ILRG50  =       50              ;ILLEGAL REGISTER 50
1177            000052          ILRG52  =       52              ;ILLEGAL REGISTER 52
1178            000054          ILRG54  =       54              ;ILLEGAL REGISTER 54
1179            000056          ILRG56  =       56              ;ILLEGAL REGISTER 56
1180            000060          ILRG60  =       60              ;ILLEGAL REGISTER 60
1181            000062          ILRG62  =       62              ;ILLEGAL REGISTER 62
1182            000064          ILRG64  =       64              ;ILLEGAL REGISTER 64
1183            000066          ILRG66  =       66              ;ILLEGAL REGISTER 66
1184            000070          ILRG70  =       70              ;ILLEGAL REGISTER 70
1185            000072          ILRG72  =       72              ;ILLEGAL REGISTER 72
1186            000074          ILRG74  =       74              ;ILLEGAL REGISTER 74
1187            000076          ILRG76  =       76              ;ILLEGAL REGISTER 76
1188
1189
1190            000077          IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
1191
```

```
                        .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER #1

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

                        ;RMCS2  RH CONTROL STATUS REGISTER #2

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

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

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

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

```
1248            000022                  RMOB    =       22              ;DATA BUFFER
1249            000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
1250            000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3
1251
1252            176700                  ABASE   =       176700          ;UNIBUS ADDRESS
1253            120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY
1254
1255
1256                                    .SBTTL  TRAP CATCHER
1257
1258            000000                          .=0
1259                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1260                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1261                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1262            000174                          .=174
1263  000174    000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
1264  000176    000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
1265
1266
1267                                    .SBTTL  ACT11 HOOKS
1268
1269                                    ;;*****************************************************************
1270                                    ;HOOKS REQUIRED BY ACT11
1271            000200                          $SVPC=.                 ;SAVE PC
1272            000046                          .=46
1273  000046    033156                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1274            000052                          .=52
1275  000052    000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
1276            000200                          .=$SVPC                 ;; RESTORE PC
1277
1278                                    .SBTTL  STARTING ADDRESS
1279
1280                                    ;THE PROGRAM STARTS AT LOCATION 200
1281            000200                  .       =       200
1282  000200    000137  005324                  JMP     START           ;JUMP TO START OF PROGRAM
1283
1284            001100                  .=1100
1285                                    .SBTTL  APT PARAMETER BLOCK
1286
1287                                    ;;*****************************************************************
1288                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1289                                    ;;*****************************************************************
1290            001100                          .$X=.   ;;SAVE CURRENT LOCATION
1291            000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1292  000024    000200                          200     ;;FOR APT START UP
1293            000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1294  000044    001100                          $APTHDR ;;POINT TO APT HEADER BLOCK
1295            001100                          .=.$X   ;;RESET LOCATION COUNTER
1296                                    ;;*****************************************************************
1297                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1298                                    ;INTERFACE SPEC.
1299
1300  001100                            $APTHD:
1301  001100    000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1302  001102    001222                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1303  001104    000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

```
  1304  001106  000002                SPASTM: .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  1305  001110  000002                SUNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  1306  001112  000042                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  1307          001114                TAGADR  =       .
```

```
                                .SBTTL  COMMON TAGS

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

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

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

```
1364  001226  000000                  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
1365  001230  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
1366  001232  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1367  001234  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1368  001236  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1369  001240  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1370  001242                          $ETABLE:                ;;APT ENVIRONMENT TABLE
1371  001242      000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1372  001243      000                 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1373  001244  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1374  001246  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1375  001250  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1376                                  ;*                      BITS 15-11=CPU TYPE
1377                                  ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1378                                  ;*                              11/70=06,PDQ=07,Q=10
1379                                  ;*                      BIT 10=REAL TIME CLOCK
1380                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
1381                                  ;*                      BIT  8=MEMORY MANAGEMENT
1382  001252      000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1383  001253      000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1384                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1385                                  ;*                              900 NSEC CORE=001
1386                                  ;*                              300 NSEC BIPOLAR=002
1387                                  ;*                              500 NSEC MOS=003
1388  001254  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1389                                  ;*                      MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1390  001256      000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1391  001257      000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1392  001260  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1393  001262      000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1394  001263      000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1395  001264  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1396  001266      000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1397  001267      000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1398  001270  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1399  001272  120254                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1400  001274  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1401  001276  176700                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1402  001300  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1403  001302  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1404  001304  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1405  001306  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
1406  001310  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
1407  001312  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
1408  001314  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
1409  001316  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
1410  001320  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
1411  001322  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1412  001324  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1413  001326                          $ETEND:
1414                                  .MEXIT
1415  001326  000000                  CTLFG:  .WORD   0
1416
1417                                  ;THE REGISTER INPUT BUFFER IS USED FOR
1418                                  ;STORING DRIVE STATUS
1419
```

```
001330                          GETBUF:

                                ;REGISTER INPUT BUFFER
001330  000000                  RMCS1I: .WORD                   ;CONTROL STATUS REGISTER
001332  000000                  RMWCI: .WORD                    ;WORD COUNT REGISTER
001334  000000                  RMBAI: .WORD                    ;BUS ADDRESS REGISTER
001336  000000                  RMDAI: .WORD                    ;DISK ADDRESS REGISTER
001340  000000                  RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
001342  000000                  RMDSI: .WORD                    ;DRIVE STATUS REGISTER
001344  000000                  RMER1I: .WORD                   ;ERROR REGISTER 1
001346  000000                  RMASI: .WORD                    ;ATTENTION SUMMARY REGISTER
001350  000000                  RMLAI: .WORD                    ;LOOK AHEAD REGISTER
001352  000000                  RMDBI: .WORD                    ;DATA BUFFER
001354  000000                  RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
001356  000000                  RMDTI: .WORD                    ;DRIVE TYPE REGISTER
001360  000000                  RMSNI: .WORD                    ;SERIAL NUMBER REGISTER
001362  000000                  RMOFI: .WORD                    ;OFFSET REGISTER
001364  000000                  RMDCI: .WORD                    ;DESIRED CYLINDER REGISTER
001366  000000                  RMCCI: .WORD                    ;CURRENT CYLINDER REGISTER
001370  000000                  RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
001372  000000                  RMER2I: .WORD                   ;ERROR REGISTER 2
001374  000000                  RMEC1I: .WORD                   ;ECC POSITION REGISTER
001376  000000                  RMEC2I: .WORD                   ;ECC PATTERN REGISTER

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

001400                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001400  000000                  RMCS1O: .WORD                   ;CONTROL STATUS REGISTER
001402  000000                  RMWCO: .WORD                    ;WORD COUNT REGISTER
001404  000000                  RMBAO: .WORD                    ;BUS ADDRESS REGISTER
001406  000000                  RMDAO: .WORD                    ;DISK ADDRESS REGISTER
001410  000000                  RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
001412  000000                  RMDSO: .WORD                    ;DRIVE STATUS REGISTER
001414  000000                  RMER1O: .WORD                   ;ERROR REGISTER 1
001416  000000                  RMASO: .WORD                    ;ATTENTION SUMMARY REGISTER
001420  000000                  RMLAO: .WORD                    ;LOOK AHEAD REGISTER
001422  000000                  RMDBO: .WORD                    ;DATA BUFFER
001424  000000                  RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
001426  000000                  RMDTO: .WORD                    ;DRIVE TYPE REGISTER
001430  000000                  RMSNO: .WORD                    ;SERIAL NUMBER REGISTER
001432  000000                  RMOFO: .WORD                    ;OFFSET REGISTER
001434  000000                  RMDCO: .WORD                    ;DESIRED CYLINDER REGISTER
001436  000000                  RMCCO: .WORD                    ;CURRENT CYLINDER REGISTER
001440  000000                  RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
001442  000000                  RMER2O: .WORD                   ;ERROR REGISTER 2
001444  000000                  RMEC1O: .WORD                   ;ECC POSITION REGISTER
001446  000000                  RMEC2O: .WORD                   ;ECC PATTERN REGISTER

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

001450  000012                  TSTQUE: .BLKW   10.      ;TEST QUE
```

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

```
1500                                    .SBTTL  ERROR POINTER TABLE
1501
1502                                    ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1503                                    ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1504                                    ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1505                                    ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1506                                    ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1507
1508                                    ;*      EM              ;;POINTS TO THE ERROR MESSAGE
1509                                    ;*      DH              ;;POINTS TO THE DATA HEADER
1510                                    ;*      DT              ;;POINTS TO THE DATA
1511                                    ;*      DF              ;;POINTS TO THE DATA FORMAT
1512
1513
1514  001564                            $ERRTB:
1515
1516
```

```
1517                                    ;ERROR  1       WRONG UNIT SELECTED
1518  001564  065432                            EMT1
1519  001566  071524                            EHT1
1520  001570  071650                            EDT1
1521  001572  071740                            EFT1
1522
1523
1524                                    ;ERROR  2       DEVICE WENT UNAVAILABLE
1525  001574  065436                            EMT2
1526  001576  071524                            EHT1
1527  001600  071650                            EDT1
1528  001602  071740                            EFT1
1529
1530
1531                                    ;ERROR  3       DEVICE WENT NONEXISTENT
1532  001604  065444                            EMT3
1533  001606  071524                            EHT1
1534  001610  071650                            EDT1
1535  001612  071740                            EFT1
1536
1537
1538                                    ;ERROR  4       CONTROLLER NOT READY
1539  001614  065452                            EMT4
1540  001616  071524                            EHT1
1541  001620  071650                            EDT1
1542  001622  071740                            EFT1
1543
1544
1545                                    ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1546  001624  065460                            EMT5
1547  001626  071524                            EHT1
1548  001630  071650                            EDT1
1549  001632  071740                            EFT1
1550
1551
1552                                    ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1553  001634  065466                            EMT6
1554  001636  071524                            EHT1
1555  001640  071650                            EDT1
1556  001642  071740                            EFT1
1557
1558
1559                                    ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1560  001644  065474                            EMT7
1561  001646  000000                            0
1562  001650  000000                            0
1563  001652  000000                            0
1564
1565
1566                                    ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
1567  001654  065502                            EMT10
1568  001656  071524                            EHT1
1569  001660  071650                            EDT1
1570  001662  071740                            EFT1
1571
1572
```

```
1573
1574  001664  065506                ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
1575  001666  071524                        EMT11
1576  001670  071650                        EHT1
1577  001672  071740                        EDT1
1578                                  EFT1
1579
1580                                ;ERROR  12      INCORRECT FUNCTION CODE
1581  001674  065512                        EMT12
1582  001676  071524                        EHT1
1583  001700  071650                        EDT1
1584  001702  071740                        EFT1
1585
1586
1587                                ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1588  001704  065520                        EMT13
1589  001706  071524                        EHT1
1590  001710  071650                        EDT1
1591  001712  071740                        EFT1
1592
1593
1594                                ;ERROR  14      TRANSFER ERROR IS INCORRECT
1595  001714  065532                        EMT14
1596  001716  071524                        EHT1
1597  001720  071650                        EDT1
1598  001722  071740                        EFT1
1599
1600
1601                                ;ERROR  15      INCORRECT WORD COUNT
1602  001724  065540                        EMT15
1603  001726  071524                        EHT1
1604  001730  071650                        EDT1
1605  001732  071740                        EFT1
1606
1607
1608                                ;ERROR  16      INCORRECT BUS ADDRESS
1609  001734  065546                        EMT16
1610  001736  071524                        EHT1
1611  001740  071650                        EDT1
1612  001742  071740                        EFT1
1613
1614
1615                                ;ERROR  17      INCORRECT LBT STATUS
1616  001744  065556                        EMT17
1617  001746  071524                        EHT1
1618  001750  071650                        EDT1
1619  001752  071740                        EFT1
1620
1621
1622                                ;ERROR  20      INCORRECT AOE
1623  001754  065566                        EMT20
1624  001756  071524                        EHT1
1625  001760  071650                        EDT1
1626  001762  071740                        EFT1
1627
1628
```

```
1629
1630  001764  065576                    ;ERROR  21      INCORRECT DISK ADDRESS
1631  001766  071524                            EMT21
1632  001770  071650                            EHT1
1633  001772  071740                            EDT1
1634                                      EFT1
1635
1636                                      ;ERROR  22      INCORRECT CYLINDER ADDRESS
1637  001774  065606                            EMT22
1638  001776  071524                            EHT1
1639  002000  071650                            EDT1
1640  002002  071740                            EFT1
1641
1642
1643                                      ;ERROR  23      INCORRECT WLE STATUS
1644  002004  065616                            EMT23
1645  002006  071524                            EHT1
1646  002010  071650                            EDT1
1647  002012  071740                            EFT1
1648
1649
1650                                      ;ERROR  24      INCORRECT UPE STATUS
1651  002014  065626                            EMT24
1652  002016  071524                            EHT1
1653  002020  071650                            EDT1
1654  002022  071740                            EFT1
1655
1656
1657                                      ;ERROR  25      INCORRECT WCF STATUS
1658  002024  065636                            EMT25
1659  002026  071524                            EHT1
1660  002030  071650                            EDT1
1661  002032  071740                            EFT1
1662
1663
1664                                      ;ERROR  26      INCORRECT WCE STATUS
1665  002034  065646                            EMT26
1666  002036  071524                            EHT1
1667  002040  071650                            EDT1
1668  002042  071740                            EFT1
1669
1670
1671                                      ;ERROR  27      INCORRECT MDPE STATUS
1672  002044  065656                            EMT27
1673  002046  071524                            EHT1
1674  002050  071650                            EDT1
1675  002052  071740                            EFT1
1676
1677
1678                                      ;ERROR  30      INCORRECT DCK STATUS
1679  002054  065666                            EMT30
1680  002056  071524                            EHT1
1681  002060  071650                            EDT1
1682  002062  071740                            EFT1
1683
1684
```

```
1685                                    ;ERROR  31      INCORRECT ECH STATUS
1686  002064  065676                            EMT31
1687  002066  071524                            EHT1
1688  002070  071650                            EDT1
1689  002072  071740                            EFT1
1690
1691
1692                                    ;ERROR  32      DLT SHOULD NOT BE SET
1693  002074  065706                            EMT32
1694  002076  071524                            EHT1
1695  002100  071650                            EDT1
1696  002102  071740                            EFT1
1697
1698
1699                                    ;ERROR  33      MXF SHOULD NOT BE SET
1700  002104  065716                            EMT33
1701  002106  071524                            EHT1
1702  002110  071650                            EDT1
1703  002112  071740                            EFT1
1704
1705
1706                                    ;ERROR  34      DTE SHOULD NOT BE SET
1707  002114  065726                            EMT34
1708  002116  071524                            EHT1
1709  002120  071650                            EDT1
1710  002122  071740                            EFT1
1711
1712
1713                                    ;ERROR  35      INCORRECT HCRC STATUS
1714  002124  065736                            EMT35
1715  002126  071524                            EHT1
1716  002130  071650                            EDT1
1717  002132  071740                            EFT1
1718
1719
1720                                    ;ERROR  36      INCORRECT HCE STATUS
1721  002134  065746                            EMT36
1722  002136  071524                            EHT1
1723  002140  071650                            EDT1
1724  002142  071740                            EFT1
1725
1726
1727                                    ;ERROR  37      INCORRECT FER STATUS
1728  002144  065756                            EMT37
1729  002146  071524                            EHT1
1730  002150  071650                            EDT1
1731  002152  071740                            EFT1
1732
1733
1734                                    ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1735  002154  065766                            EMT40
1736  002156  071524                            EHT1
1737  002160  071650                            EDT1
1738  002162  071740                            EFT1
1739
1740
```

```
1741
1742  002164  065774                ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1743  002166  071524                        EMT41
1744  002170  071650                        EHT1
1745  002172  071740                        EDT1
1746                                  EFT1
1747
1748                                ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1749  002174  066004                        EMT42
1750  002176  071524                        EHT1
1751  002200  071650                        EDT1
1752  002202  071740                        EFT1
1753
1754
1755                                ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1756  002204  066016                        EMT43
1757  002206  071524                        EHT1
1758  002210  071650                        EDT1
1759  002212  071740                        EFT1
1760
1761
1762                                ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1763  002214  066026                        EMT44
1764  002216  071524                        EHT1
1765  002220  071650                        EDT1
1766  002222  071740                        EFT1
1767
1768
1769                                ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1770  002224  066036                        EMT45
1771  002226  071524                        EHT1
1772  002230  071650                        EDT1
1773  002232  071740                        EFT1
1774
1775
1776                                ;ERROR  46      "ILF" EPROR DURING PACK ACKNOWLEDGE
1777  002234  066046                        EMT46
1778  002236  071524                        EHT1
1779  002240  071650                        EDT1
1780  002242  071740                        EFT1
1781
1782
1783                                ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1784  002244  066056                        EMT47
1785  002246  071524                        EHT1
1786  002250  071650                        EDT1
1787  002252  071740                        EFT1
1788
1789
1790                                ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1791  002254  066064                        EMT50
1792  002256  071524                        EHT1
1793  002260  071650                        EDT1
1794  002262  071740                        EFT1
1795
1796
```

```
1797                                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1798  002264  066074                              EMT51
1799  002266  071524                              EHT1
1800  002270  071650                              EDT1
1801  002272  071740                              EFT1
1802
1803
1804                                    ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1805  002274  066106                              EMT52
1806  002276  071524                              EHT1
1807  002300  071650                              EDT1
1808  002302  071740                              EFT1
1809
1810
1811                                    ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1812                                    ;               ON CYLINDER LATCH DIDN'T RESET
1813  002304  066124                              EMT53
1814  002306  071524                              EHT1
1815  002310  071650                              EDT1
1816  002312  071740                              EFT1
1817
1818
1819                                    ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1820  002314  066142                              EMT54
1821  002316  071524                              EHT1
1822  002320  071650                              EDT1
1823  002322  071740                              EFT1
1824
1825
1826                                    ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1827  002324  066152                              EMT55
1828  002326  071524                              EHT1
1829  002330  071650                              EDT1
1830  002332  071740                              EFT1
1831
1832
1833                                    ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1834  002334  066164                              EMT56
1835  002336  071524                              EHT1
1836  002340  071650                              EDT1
1837  002342  071740                              EFT1
1838
1839
1840                                    ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1841  002344  066202                              EMT57
1842  002346  071524                              EHT1
1843  002350  071650                              EDT1
1844  002352  071740                              EFT1
1845
1846
1847                                    ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1848                                    ;               VOLUME VALID
1849  002354  066212                              EMT60
1850  002356  071524                              EHT1
1851  002360  071650                              EDT1
1852  002362  071740                              EFT1
```

```
                                ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
                                ;               VOLUME VALID IS STIL SET
002364  066230                          EMT61
002366  071524                          EHT1
002370  071650                          EDT1
002372  071740                          EFT1


                                ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
                                ;               REPORTED DURING SEEK COMMAND
002374  066250                          EMT62
002376  071524                          EHT1
002400  071650                          EDT1
002402  071740                          EFT1


                                ;ERROR  63      UNUSED
002404  000000                          0
002406  000000                          0
002410  000000                          0
002412  000000                          0


                                ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
002414  066266                          EMT64
002416  071524                          EHT1
002420  071650                          EDT1
002422  071740                          EFT1


                                ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
002424  066306                          EMT65
002426  071524                          EHT1
002430  071650                          EDT1
002432  071740                          EFT1


                                ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
002434  066326                          EMT66
002436  071524                          EHT1
002440  071650                          EDT1
002442  071740                          EFT1


                                ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
002444  066340                          EMT67
002446  071524                          EHT1
002450  071650                          EDT1
002452  071740                          EFT1


                                ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
002454  066352                          EMT70
002456  071524                          EHT1
```

```
 1909  002460  071650                        EDT1
 1910  002462  071740                        EFT1
 1911
 1912
 1913                          ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
 1914  002464  066362                        EMT71
 1915  002466  071524                        EHT1
 1916  002470  071650                        EDT1
 1917  002472  071740                        EFT1
 1918
 1919
 1920                          ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
 1921  002474  066372                        EMT72
 1922  002476  071524                        EHT1
 1923  002500  071650                        EDT1
 1924  002502  071740                        EFT1
 1925
 1926
 1927                          ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
 1928                          ;               CYLINDER DIDNT DROP
 1929  002504  066410                        EMT73
 1930  002506  071524                        EHT1
 1931  002510  071650                        EDT1
 1932  002512  071740                        EFT1
 1933
 1934
 1935                          ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
 1936  002514  066426                        EMT74
 1937  002516  071524                        EHT1
 1938  002520  071650                        EDT1
 1939  002522  071740                        EFT1
 1940
 1941
 1942
 1943  002524  066436                        EMT75
 1944  002526  071524                        EHT1
 1945  002530  071650                        EDT1
 1946  002532  071740                        EFT1
 1947
 1948
 1949                          ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
 1950  002534  066456                        EMT76
 1951  002536  071524                        EHT1
 1952  002540  071650                        EDT1
 1953  002542  071740                        EFT1
 1954
 1955
 1956                          ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
 1957  002544  066466                        EMT77
 1958  002546  071524                        EHT1
 1959  002550  071650                        EDT1
 1960  002552  071740                        EFT1
 1961
 1962
 1963                          ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
 1964  002554  066500                        EMT100
```

```
 1965  002556  071524                          EHT1
 1966  002560  071650                          EDT1
 1967  002562  071740                          EFT1
 1968
 1969
 1970                                ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
 1971  002564  066516                          EMT101
 1972  002566  071524                          EHT1
 1973  002570  071650                          EDT1
 1974  002572  071740                          EFT1
 1975
 1976
 1977                                ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
 1978  002574  066534                          EMT102
 1979  002576  071524                          EHT1
 1980  002600  071650                          EDT1
 1981  002602  071740                          EFT1
 1982
 1983
 1984                                ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
 1985  002604  066544                          EMT103
 1986  002606  071524                          EHT1
 1987  002610  071650                          EDT1
 1988  002612  071740                          EFT1
 1989
 1990
 1991                                ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
 1992  002614  066556                          EMT104
 1993  002616  071524                          EHT1
 1994  002620  071650                          EDT1
 1995  002622  071740                          EFT1
 1996
 1997
 1998                                ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
 1999  002624  066574                          EMT105
 2000  002626  071524                          EHT1
 2001  002630  071650                          EDT1
 2002  002632  071740                          EFT1
 2003
 2004
 2005                                ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
 2006  002634  066604                          EMT106
 2007  002636  071524                          EHT1
 2008  002640  071650                          EDT1
 2009  002642  071740                          EFT1
 2010
 2011
 2012                                ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
 2013  002644  066614                          EMT107
 2014  002646  071524                          EHT1
 2015  002650  071650                          EDT1
 2016  002652  071740                          EFT1
 2017
 2018
 2019                                ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
 2020  002654  066634                          EMT110
```

```
2021   002656  071546                  EHT110
2022   002660  071666                  EDT110
2023   002662  071756                  EFT110
2024
2025
2026                           ;ERROR  111     NONEXISTENT DEVICE
2027   002664  066646                  EMT111
2028   002666  071552                  EHT111
2029   002670  071670                  EDT111
2030   002672  071760                  EFT111
2031
2032
2033                           ;ERROR  112     DEVICE NOT AVAILABLE
2034   002674  066654                  EMT112
2035   002676  071552                  EHT111
2036   002700  071670                  EDT111
2037   002702  071760                  EFT111
2038
2039
2040                           ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
2041   002704  066662                  EMT113
2042   002706  000000                  0
2043   002710  000000                  0
2044   002712  000000                  0
2045
2046
2047                           ;ERROR  114     DEVICE NOT AN RM03
2048   002714  066676                  EMT114
2049   002716  071556                  EHT114
2050   002720  071672                  EDT114
2051   002722  071762                  EFT114
2052
2053
2054                           ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
2055   002724  066704                  EMT115
2056   002726  071524                  EHT1
2057   002730  071650                  EDT1
2058   002732  071740                  EFT1
2059
2060
2061                           ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
2062   002734  066714                  EMT116
2063   002736  071524                  EHT1
2064   002740  071650                  EDT1
2065   002742  071740                  EFT1
2066
2067
2068                           ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
2069   002744  066724                  EMT117
2070   002746  071524                  EHT1
2071   002750  071650                  EDT1
2072   002752  071740                  EFT1
2073
2074
2075                           ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
2076   002754  066734                  EMT120
```

```
2077  002756  071524                        EHT1
2078  002760  071650                        EDT1
2079  002762  071740                        EFT1
2080
2081
2082                                ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
2083  002764  066744                        EMT121
2084  002766  071524                        EHT1
2085  002770  071650                        EDT1
2086  002772  071740                        EFT1
2087
2088
2089                                ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
2090  002774  066754                        EMT122
2091  002776  071524                        EHT1
2092  003000  071650                        EDT1
2093  003002  071740                        EFT1
2094
2095
2096                                ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
2097  003004  066764                        EMT123
2098  003006  071524                        EHT1
2099  003010  071650                        EDT1
2100  003012  071740                        EFT1
2101
2102
2103                                ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
2104  003014  066774                        EMT124
2105  003016  071524                        EHT1
2106  003020  071650                        EDT1
2107  003022  071740                        EFT1
2108
2109
2110
2111  003024  067004                ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
2112  003026  071524                        EMT125
2113  003030  071650                        EHT1
2114  003032  071740                        EDT1
2115                                        EFT1
2116
2117                                ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2118  003034  067014                        EMT126
2119  003036  071524                        EHT1
2120  003040  071650                        EDT1
2121  003042  071740                        EFT1
2122
2123
2124                                ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
2125  003044  067026                        EMT127
2126  003046  071524                        EHT1
2127  003050  071650                        EDT1
2128  003052  071740                        EFT1
2129
2130
2131                                ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2132  003054  067040                        EMT130
```

```
2133  003056  071524                        EHT1
2134  003060  071650                        EDT1
2135  003062  071740                        EFT1
2136
2137
2138                                ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2139  003064  067052                        EMT131
2140  003066  071524                        EHT1
2141  003070  071650                        EDT1
2142  003072  071740                        EFT1
2143
2144
2145                                ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2146  003074  067064                        EMT132
2147  003076  071524                        EHT1
2148  003100  071650                        EDT1
2149  003102  071740                        EFT1
2150
2151
2152                                ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2153  003104  067076                        EMT133
2154  003106  071524                        EHT1
2155  003110  071650                        EDT1
2156  003112  071740                        EFT1
2157
2158
2159                                ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2160  003114  067110                        EMT134
2161  003116  071524                        EHT1
2162  003120  071650                        EDT1
2163  003122  071740                        EFT1
2164
2165
2166                                ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2167  003124  067122                        EMT135
2168  003126  071524                        EHT1
2169  003130  071650                        EDT1
2170  003132  071740                        EFT1
2171
2172
2173                                ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2174  003134  067134                        EMT136
2175  003136  071524                        EHT1
2176  003140  071650                        EDT1
2177  003142  071740                        EFT1
2178
2179
2180                                ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2181  003144  067146                        EMT137
2182  003146  071524                        EHT1
2183  003150  071650                        EDT1
2184  003152  071740                        EFT1
2185
2186
2187                                ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2188  003154  067156                        EMT140
```

```
 2189  003156  071524                    EHT1
 2190  003160  071650                    EDT1
 2191  003162  071740                    EFT1
 2192
 2193
 2194                          ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
 2195  003164  067166                    EMT141
 2196  003166  071524                    EHT1
 2197  003170  071650                    EDT1
 2198  003172  071740                    EFT1
 2199
 2200
 2201                          ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
 2202  003174  067176                    EMT142
 2203  003176  071524                    EHT1
 2204  003200  071650                    EDT1
 2205  003202  071740                    EFT1
 2206
 2207
 2208                          ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
 2209  003204  067206                    EMT143
 2210  003206  071524                    EHT1
 2211  003210  071650                    EDT1
 2212  003212  071740                    EFT1
 2213
 2214
 2215                          ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
 2216  003214  067216                    EMT144
 2217  003216  071524                    EHT1
 2218  003220  071650                    EDT1
 2219  003222  071740                    EFT1
 2220
 2221
 2222
 2223  003224  067226                    EMT145
 2224  003226  071524                    EHT1
 2225  003230  071650                    EDT1
 2226  003232  071740                    EFT1
 2227
 2228
 2229                          ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
 2230  003234  067236                    EMT146
 2231  003236  071524                    EHT1
 2232  003240  071650                    EDT1
 2233  003242  071740                    EFT1
 2234
 2235
 2236                          ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
 2237  003244  067246                    EMT147
 2238  003246  071524                    EHT1
 2239  003250  071650                    EDT1
 2240  003252  071740                    EFT1
 2241
 2242
 2243                          ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
 2244  003254  067256                    EMT150
```

```
2245  003256  071524                    EHT1
2246  003260  071650                    EDT1
2247  003262  071740                    EFT1
2248
2249
2250                                ;ERROR  151     MEDIUM NOT ON LINE
2251  003264  067266                    EMT151
2252  003266  071524                    EHT1
2253  003270  071650                    EDT1
2254  003272  071740                    EFT1
2255
2256
2257                                ;ERROR  152     DRIVE FAULT
2258  003274  067300                    EMT152
2259  003276  071524                    EHT1
2260  003300  071650                    EDT1
2261  003302  071740                    EFT1
2262
2263
2264                                ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2265  003304  067312                    EMT153
2266  003306  071524                    EHT1
2267  003310  071650                    EDT1
2268  003312  071740                    EFT1
2269
2270
2271                                ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2272  003314  067330                    EMT154
2273  003316  071524                    EHT1
2274  003320  071650                    EDT1
2275  003322  071740                    EFT1
2276
2277
2278                                ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2279  003324  067346                    EMT155
2280  003326  071524                    EHT1
2281  003330  071650                    EDT1
2282  003332  071740                    EFT1
2283
2284
2285                                ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2286  003334  067360                    EMT156
2287  003336  071524                    EHT1
2288  003340  071650                    EDT1
2289  003342  071740                    EFT1
2290
2291
2292                                ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2293  003344  067372                    EMT157
2294  003346  071524                    EHT1
2295  003350  071650                    EDT1
2296  003352  071740                    EFT1
2297
2298
2299                                ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2300  003354  067404                    EMT160
```

```
2301   003356  071524                          EHT1
2302   003360  071650                          EDT1
2303   003362  071740                          EFT1
2304
2305
2306                                   ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2307   003364  067414                          EMT161
2308   003366  071524                          EHT1
2309   003370  071650                          EDT1
2310   003372  071740                          EFT1
2311
2312
2313                                   ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2314   003374  067426                          EMT162
2315   003376  071524                          EHT1
2316   003400  071650                          EDT1
2317   003402  071740                          EFT1
2318
2319
2320                                   ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2321                                   ;               WRITE BUFFER
2322   003404  071320                          EMT336
2323   003406  071606                          EHT336
2324   003410  071704                          EDT336
2325   003412  071774                          EFT336
2326
2327
2328                                   ;ERROR  164     OPI SHOULD NOT BE SET
2329   003414  067450                          EMT164
2330   003416  071524                          EHT1
2331   003420  071650                          EDT1
2332   003422  071740                          EFT1
2333
2334
2335                                   ;ERROR  165     IVC SHOULD NOT BE SET
2336   003424  067456                          EMT165
2337   003426  071524                          EHT1
2338   003430  071650                          EDT1
2339   003432  071740                          EFT1
2340
2341
2342                                   ;ERROR  166     IAE SHOULD NOT BE SET
2343   003434  067464                          EMT166
2344   003436  071524                          EHT1
2345   003440  071650                          EDT1
2346   003442  071740                          EFT1
2347
2348
2349                                   ;ERROR  167     NEM SHOULD NOT BE SET
2350   003444  067472                          EMT167
2351   003446  071524                          EHT1
2352   003450  071650                          EDT1
2353   003452  071740                          EFT1
2354
2355
2356                                   ;ERROR 170      UNUSED
```

```
2357  003454  000000                    0
2358  003456  000000                    0
2359  003460  000000                    0
2360  003462  000000                    0
2361
2362
2363                          ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2364  003464  067510                    EMT171
2365  003466  071524                    EHT1
2366  003470  071650                    EDT1
2367  003472  071740                    EFT1
2368
2369
2370                          ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2371  003474  067520                    EMT172
2372  003476  071524                    EHT1
2373  003500  071650                    EDT1
2374  003502  071740                    EFT1
2375
2376
2377                          ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2378  003504  067530                    EMT173
2379  003506  071524                    EHT1
2380  003510  071650                    EDT1
2381  003512  071740                    EFT1
2382
2383
2384                          ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2385  003514  067540                    EMT174
2386  003516  071524                    EHT1
2387  003520  071650                    EDT1
2388  003522  071740                    EFT1
2389
2390
2391                          ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2392  003524  067552                    EMT175
2393  003526  071524                    EHT1
2394  003530  071650                    EDT1
2395  003532  071740                    EFT1
2396
2397
2398                          ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2399  003534  067560                    EMT176
2400  003536  071524                    EHT1
2401  003540  071650                    EDT1
2402  003542  071740                    EFT1
2403
2404
2405                          ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2406  003544  067566                    EMT177
2407  003546  071524                    EHT1
2408  003550  071650                    EDT1
2409  003552  071740                    EFT1
2410
2411
2412                          ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
2413  003554  067600                    EMT200
2414  003556  071524                    EHT1
2415  003560  071650                    EDT1
2416  003562  071740                    EFT1
2417
2418
2419                              ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2420  003564  067612                    EMT201
2421  003566  071524                    EHT1
2422  003570  071650                    EDT1
2423  003572  071740                    EFT1
2424
2425
2426                              ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2427  003574  067624                    EMT202
2428  003576  071524                    EHT1
2429  003600  071650                    EDT1
2430  003602  071740                    EFT1
2431
2432
2433                              ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2434  003604  067636                    EMT203
2435  003606  071524                    EHT1
2436  003610  071650                    EDT1
2437  003612  071740                    EFT1
2438
2439
2440                              ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2441  003614  067650                    EMT204
2442  003616  071524                    EHT1
2443  003620  071650                    EDT1
2444  003622  071740                    EFT1
2445
2446
2447                              ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2448  003624  067666                    EMT205
2449  003626  071524                    EHT1
2450  003630  071650                    EDT1
2451  003632  071740                    EFT1
2452
2453
2454                              ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2455  003634  067676                    EMT206
2456  003636  071524                    EHT1
2457  003640  071650                    EDT1
2458  003642  071740                    EFT1
2459
2460
2461                              ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2462  003644  067706                    EMT207
2463  003646  071524                    EHT1
2464  003650  071650                    EDT1
2465  003652  071740                    EFT1
2466
2467
2468                              ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2469  003654  067720                    EMT210
2470  003656  071524                    EHT1
2471  003660  071650                    EDT1
2472  003662  071740                    EFT1
2473
2474
2475                              ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2476  003664  067726                    EMT211
2477  003666  071524                    EHT1
2478  003670  071650                    EDT1
2479  003672  071740                    EFT1
2480
2481
2482                              ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2483  003674  067742                    EMT212
2484  003676  071524                    EHT1
2485  003700  071650                    EDT1
2486  003702  071740                    EFT1
2487
2488
2489                              ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2490  003704  067756                    EMT213
2491  003706  071524                    EHT1
2492  003710  071650                    EDT1
2493  003712  071740                    EFT1
2494
2495
2496                              ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2497  003714  067766                    EMT214
2498  003716  071536                    EHT2
2499  003720  071660                    EDT2
2500  003722  071750                    EFT2
2501
2502
2503                              ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2504  003724  070006                    EMT215
2505  003726  071536                    EHT2
2506  003730  071660                    EDT2
2507  003732  071750                    EFT2
2508
2509
2510                              ;ERROR  216     INCORRECT "IVC" STATUS
2511  003734  070020                    EMT216
2512  003736  071524                    EHT1
2513  003740  071650                    EDT1
2514  003742  071740                    EFT1
2515
2516
2517                              ;ERROR  217     INCORRECT "IAE" STATUS
2518  003744  070030                    EMT217
2519  003746  071524                    EHT1
2520  003750  071650                    EDT1
2521  003752  071740                    EFT1
2522
2523
2524                              ;ERROR 220      INCORRECT "WLE" STATUS
```

```
2525  003754  070040                        EMT220
2526  003756  071524                        EHT1
2527  003760  071650                        EDT1
2528  003762  071740                        EFT1
2529
2530
2531                                ;ERROR  221     INCORRECT "OPI" STATUS
2532  003764  070050                        EMT221
2533  003766  071524                        EHT1
2534  003770  071650                        EDT1
2535  003772  071740                        EFT1
2536
2537
2538                                ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2539  003774  070060                        EMT222
2540  003776  071524                        EHT1
2541  004000  071650                        EDT1
2542  004002  071740                        EFT1
2543
2544
2545                                ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2546  004004  070070                        EMT223
2547  004006  071562                        EHT223
2548  004010  071674                        EDT223
2549  004012  071764                        EFT223
2550
2551
2552                                ;ERROR  224     UNUSED
2553  004014  000000                        0
2554  004016  000000                        0
2555  004020  000000                        0
2556  004022  000000                        0
2557
2558
2559                                ;ERROR  225     UNUSED
2560  004024  000000                        0
2561  004026  000000                        0
2562  004030  000000                        0
2563  004032  000000                        0
2564
2565
2566                                ;ERROR  226     UNUSED
2567  004034  000000                        0
2568  004036  000000                        0
2569  004040  000000                        0
2570  004042  000000                        0
2571
2572
2573                                ;ERROR  227     UNUSED
2574  004044  000000                        0
2575  004046  000000                        0
2576  004050  000000                        0
2577  004052  000000                        0
2578
2579
2580                                ;ERROR  230     UNUSED
```

```
2581  004054  000000                    0
2582  004056  000000                    0
2583  004060  000000                    0
2584  004062  000000                    0
2585
2586
2587                              ;ERROR  231     UNUSED
2588  004064  000000                    0
2589  004066  000000                    0
2590  004070  000000                    0
2591  004072  000000                    0
2592
2593
2594                              ;ERROR  232     UNUSED
2595  004074  000000                    0
2596  004076  000000                    0
2597  004100  000000                    0
2598  004102  000000                    0
2599
2600
2601                              ;ERROR  233     UNUSED
2602  004104  000000                    0
2603  004106  000000                    0
2604  004110  000000                    0
2605  004112  000000                    0
2606
2607
2608                              ;ERROR  234     UNUSED
2609  004114  000000                    0
2610  004116  000000                    0
2611  004120  000000                    0
2612  004122  000000                    0
2613
2614
2615                              ;ERROR  235     UNUSED
2616  004124  000000                    0
2617  004126  000000                    0
2618  004130  000000                    0
2619  004132  000000                    0
2620
2621
2622                              ;ERROR  236     UNUSED
2623  004134  000000                    0
2624  004136  000000                    0
2625  004140  000000                    0
2626  004142  000000                    0
2627
2628
2629                              ;ERROR  237     UNUSED
2630  004144  000000                    0
2631  004146  000000                    0
2632  004150  000000                    0
2633  004152  000000                    0
2634
2635
2636                              ;ERROR  240     UNUSED
```

```
2637  004154  000000                          0
2638  004156  000000                          0
2639  004160  000000                          0
2640  004162  000000                          0
2641
2642
2643                                ;ERROR  241     UNUSED
2644  004164  000000                          0
2645  004166  000000                          0
2646  004170  000000                          0
2647  004172  000000                          0
2648
2649
2650                                ;ERROR  242     UNUSED
2651  004174  000000                          0
2652  004176  000000                          0
2653  004200  000000                          0
2654  004202  000000                          0
2655
2656
2657                                ;ERROR  243     UNUSED
2658  004204  000000                          0
2659  004206  000000                          0
2660  004210  000000                          0
2661  004212  000000                          0
2662
2663
2664                                ;ERROR  244     UNUSED
2665  004214  000000                          0
2666  004216  000000                          0
2667  004220  000000                          0
2668  004222  000000                          0
2669
2670
2671                                ;ERROR  245     UNUSED
2672  004224  000000                          0
2673  004226  000000                          0
2674  004230  000000                          0
2675  004232  000000                          0
2676
2677
2678                                ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2679  004234  070144                          EMT246
2680  004236  071524                          EHT1
2681  004240  071650                          EDT1
2682  004242  071740                          EFT1
2683
2684
2685                                ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2686  004244  070154                          EMT247
2687  004246  071524                          EHT1
2688  004250  071650                          EDT1
2689  004252  071740                          EFT1
2690
2691
2692                                ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
2693  004254  070166                        EMT250
2694  004256  071524                        EHT1
2695  004260  071650                        EDT1
2696  004262  071740                        EFT1
2697
2698
2699                                ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2700  004264  070202                        EMT251
2701  004266  071536                        EHT2
2702  004270  071660                        EDT2
2703  004272  071750                        EFT2
2704
2705
2706                                ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2707  004274  070210                        EMT252
2708  004276  071536                        EHT2
2709  004300  071660                        EDT2
2710  004302  071750                        EFT2
2711
2712
2713                                ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2714  004304  070216                        EMT253
2715  004306  071524                        EHT1
2716  004310  071650                        EDT1
2717  004312  071740                        EFT1
2718
2719
2720                                ;ERROR  254     INCORRECT "ILF" STATUS
2721  004314  070234                        EMT254
2722  004316  071524                        EHT1
2723  004320  071650                        EDT1
2724  004322  071740                        EFT1
2725
2726
2727                                ;ERROR  255     INCORRECT "ATA" STATUS
2728  004324  070244                        EMT255
2729  004326  071524                        EHT1
2730  004330  071650                        EDT1
2731  004332  071740                        EFT1
2732
2733
2734                                ;ERROR  256     INCORRECT "ILR" STATUS
2735  004334  070254                        EMT256
2736  004336  071574                        EHT256
2737  004340  071674                        EDT223
2738  004342  071764                        EFT223
2739
2740
2741                                ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2742  004344  070264                        EMT257
2743  004346  071524                        EHT1
2744  004350  071650                        EDT1
2745  004352  071740                        EFT1
2746
2747
2748                                ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
2749  004354  070276                        EMT260
2750  004356  071524                        EHT1
2751  004360  071650                        EDT1
2752  004362  071740                        EFT1
2753
2754
2755                                ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2756  004364  070310                        EMT261
2757  004366  071524                        EHT1
2758  004370  071650                        EDT1
2759  004372  071740                        EFT1
2760
2761
2762                                ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2763  004374  070330                        EMT262
2764  004376  071524                        EHT1
2765  004400  071650                        EDT1
2766  004402  071740                        EFT1
2767
2768
2769                                ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2770  004404  070340                        EMT263
2771  004406  071524                        EHT1
2772  004410  071650                        EDT1
2773  004412  071740                        EFT1
2774
2775
2776                                ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2777  004414  070350                        EMT264
2778  004416  071524                        EHT1
2779  004420  071650                        EDT1
2780  004422  071740                        EFT1
2781
2782
2783                                ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2784  004424  070370                        EMT265
2785  004426  071524                        EHT1
2786  004430  071650                        EDT1
2787  004432  071740                        EFT1
2788
2789
2790                                ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2791  004434  070410                        EMT266
2792  004436  071524                        EHT1
2793  004440  071650                        EDT1
2794  004442  071740                        EFT1
2795
2796
2797                                ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2798                                ;               ADDRESS IS TOO LARGE
2799  004444  070422                        EMT267
2800  004446  071524                        EHT1
2801  004450  071650                        EDT1
2802  004452  071740                        EFT1
2803
2804
```

```
2805                                    ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
2806  004454  070440                              EMT270
2807  004456  071524                              EHT1
2808  004460  071650                              EDT1
2809  004462  071740                              EFT1
2810
2811
2812                                    ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2813                                    ;               DIDN'T DROP
2814  004464  070454                              EMT271
2815  004466  071524                              EHT1
2816  004470  071650                              EDT1
2817  004472  071740                              EFT1
2818
2819
2820                                    ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
2821  004474  070472                              EMT272
2822  004476  071524                              EHT1
2823  004500  071650                              EDT1
2824  004502  071740                              EFT1
2825
2826
2827                                    ;ERROR  273     PIP STIL SET AFTER SEARCH
2828  004504  070510                              EMT273
2829  004506  071524                              EHT1
2830  004510  071650                              EDT1
2831  004512  071740                              EFT1
2832
2833
2834                                    ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
2835                                    ;               REGISTERS BUT MXF DID NOT SET
2836  004514  070526                              EMT274
2837  004516  071524                              EHT1
2838  004520  071650                              EDT1
2839  004522  071740                              EFT1
2840
2841
2842                                    ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2843                                    ;               COMMAND STARTED
2844  004524  070544                              EMT275
2845  004526  071524                              EHT1
2846  004530  071650                              EDT1
2847  004532  071740                              EFT1
2848
2849
2850                                    ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2851                                    ;               ZERO
2852  004534  070556                              EMT276
2853  004536  071524                              EHT1
2854  004540  071650                              EDT1
2855  004542  071740                              EFT1
2856
2857
2858                                    ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2859                                    ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2860                                    ;               3) RUN TIMED OUT
```

```
2861   004544  070572                  EMT277
2862   004546  071524                  EHT1
2863   004550  071650                  EDT1
2864   004552  071740                  EFT1
2865
2866
2867                                   ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2868                                   ;               WAS NOT VALID
2869   004554  070610                  EMT300
2870   004556  071524                  EHT1
2871   004560  071650                  EDT1
2872   004562  071740                  EFT1
2873
2874
2875                                   ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2876                                   ;               IS VALID
2877   004564  070630                  EMT301
2878   004566  071524                  EHT1
2879   004570  071650                  EDT1
2880   004572  071740                  EFT1
2881
2882
2883                                   ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2884   004574  070652                  EMT302
2885   004576  071524                  EHT1
2886   004600  071650                  EDT1
2887   004602  071740                  EFT1
2888
2889
2890                                   ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2891   004604  070664                  EMT303
2892   004606  071524                  EHT1
2893   004610  071650                  EDT1
2894   004612  071740                  EFT1
2895
2896
2897                                   ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2898   004614  070676                  EMT304
2899   004616  071524                  EHT1
2900   004620  071650                  EDT1
2901   004622  071740                  EFT1
2902
2903
2904                                   ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2905   004624  070710                  EMT305
2906   004626  071524                  EHT1
2907   004630  071650                  EDT1
2908   004632  071740                  EFT1
2909
2910
2911                                   ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2912   004634  070722                  EMT306
2913   004636  071524                  EHT1
2914   004640  071650                  EDT1
2915   004642  071740                  EFT1
2916
```

```
2917
2918                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2919  004644  070732                              EMT307
2920  004646  071524                              EHT1
2921  004650  071650                              EDT1
2922  004652  071740                              EFT1
2923
2924
2925                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2926  004654  070752                              EMT310
2927  004656  071524                              EHT1
2928  004660  071650                              EDT1
2929  004662  071740                              EFT1
2930
2931
2932                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2933  004664  070764                              EMT311
2934  004666  071524                              EHT1
2935  004670  071650                              EDT1
2936  004672  071740                              EFT1
2937
2938
2939                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2940  004674  070776                              EMT312
2941  004676  071524                              EHT1
2942  004700  071650                              EDT1
2943  004702  071740                              EFT1
2944
2945
2946                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2947  004704  071010                              EMT313
2948  004706  071524                              EHT1
2949  004710  071650                              EDT1
2950  004712  071740                              EFT1
2951
2952
2953                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2954  004714  071030                              EMT314
2955  004716  071524                              EHT1
2956  004720  071650                              EDT1
2957  004722  071740                              EFT1
2958
2959
2960                                    ;ERROR  315     WRITE LOCK ERROR
2961  004724  071042                              EMT315
2962  004726  071524                              EHT1
2963  004730  071650                              EDT1
2964  004732  071740                              EFT1
2965
2966
2967                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2968  004734  071054                              EMT316
2969  004736  071524                              EHT1
2970  004740  071650                              EDT1
2971  004742  071740                              EFT1
2972
```

```
2973
2974                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2975  004744  071066                              EMT317
2976  004746  071524                              EHT1
2977  004750  071650                              EDT1
2978  004752  071740                              EFT1
2979
2980
2981                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2982  004754  071076                              EMT320
2983  004756  071524                              EHT1
2984  004760  071650                              EDT1
2985  004762  071740                              EFT1
2986
2987
2988                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2989  004764  071106                              EMT321
2990  004766  071524                              EHT1
2991  004770  071650                              EDT1
2992  004772  071740                              EFT1
2993
2994
2995                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
2996  004774  071116                              EMT322
2997  004776  071524                              EHT1
2998  005000  071650                              EDT1
2999  005002  071740                              EFT1
3000
3001
3002                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
3003  005004  071124                              EMT323
3004  005006  071524                              EHT1
3005  005010  071650                              EDT1
3006  005012  071740                              EFT1
3007
3008
3009                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3010  005014  071134                              EMT324
3011  005016  071524                              EHT1
3012  005020  071650                              EDT1
3013  005022  071740                              EFT1
3014
3015
3016                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3017  005024  071146                              EMT325
3018  005026  071524                              EHT1
3019  005030  071650                              EDT1
3020  005032  071740                              EFT1
3021
3022
3023                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
3024  005034  071160                              EMT326
3025  005036  071524                              EHT1
3026  005040  071650                              EDT1
3027  005042  071740                              EFT1
3028
```

```
3029
3030                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
3031  005044  071176                              EMT327
3032  005046  071524                              EHT1
3033  005050  071650                              EDT1
3034  005052  071740                              EFT1
3035
3036
3037                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3038  005054  071210                              EMT330
3039  005056  071524                              EHT1
3040  005060  071650                              EDT1
3041  005062  071740                              EFT1
3042
3043
3044                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3045  005064  071220                              EMT331
3046  005066  071524                              EHT1
3047  005070  071650                              EDT1
3048  005072  071740                              EFT1
3049
3050
3051                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3052  005074  071232                              EMT332
3053  005076  071524                              EHT1
3054  005100  071650                              EDT1
3055  005102  071740                              EFT1
3056
3057
3058                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3059  005104  071244                              EMT333
3060  005106  071524                              EHT1
3061  005110  071650                              EDT1
3062  005112  071740                              EFT1
3063
3064
3065                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3066  005114  071262                              EMT334
3067  005116  071524                              EHT1
3068  005120  071650                              EDT1
3069  005122  071740                              EFT1
3070
3071
3072                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3073  005124  071300                              EMT335
3074  005126  071524                              EHT1
3075  005130  071650                              EDT1
3076  005132  071740                              EFT1
3077
3078
3079                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3080  005134  071320                              EMT336
3081  005136  071606                              EHT336
3082  005140  071704                              EDT336
3083  005142  071774                              EFT336
3084
```

```
3085
3086                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3087  005144  071330                              EMT337
3088  005146  071620                              EHT337
3089  005150  071714                              EDT337
3090  005152  072004                              EFT337
3091
3092
3093                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3094  005154  071340                              EMT340
3095  005156  071606                              EHT336
3096  005160  071704                              EDT336
3097  005162  071774                              EFT336
3098
3099
3100                                    ,ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3101  005164  071352                              EMT341
3102  005166  071606                              EHT336
3103  005170  071704                              EDT336
3104  005172  071774                              EFT336
3105
3106
3107                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3108  005174  071360                              EMT342
3109  005176  071524                              EHT1
3110  005200  071650                              EDT1
3111  005202  071740                              EFT1
3112
3113
3114                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3115  005204  071372                              EMT343
3116  005206  071524                              EHT1
3117  005210  071650                              EDT1
3118  005212  071740                              EFT1
3119
3120
3121                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3122  005214  071404                              EMT344
3123  005216  071632                              EHT344
3124  005220  071724                              EDT344
3125  005222  072014                              EFT344
3126
3127
3128                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3129  005224  071416                              EMT345
3130  005226  071524                              EHT1
3131  005230  071650                              EDT1
3132  005232  071740                              EFT1
3133
3134
3135                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3136  005234  071426                              EMT346
3137  005236  071524                              EHT1
3138  005240  071650                              EDT1
3139  005242  071740                              EFT1
3140
```

```
3141
3142                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3143  005244  071442                              EMT347
3144  005246  071524                              EHT1
3145  005250  071650                              EDT1
3146  005252  071740                              EFT1
3147
3148
3149                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3150  005254  071454                              EMT350
3151  005256  071524                              EHT1
3152  005260  071650                              EDT1
3153  005262  071740                              EFT1
3154
3155
3156                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3157  005264  071472                              EMT351
3158  005266  071524                              EHT1
3159  005270  071650                              EDT1
3160  005272  071740                              EFT1
3161
3162
3163                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3164  005274  071502                              EMT352
3165  005276  000000                              0
3166  005300  000000                              0
3167  005302  000000                              0
3168
3169
3170                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3171  005304  071506                              EMT353
3172  005306  071644                              EHT353
3173  005310  071736                              EDT353
3174  005312  072024                              EFT353
3175
3176
3177                                    ;ERROR  354     BSE SHOULD NOT BE SET
3178  005314  071516                              EMT354
3179  005316  071524                              EHT1
3180  005320  071650                              EDT1
3181  005322  071740                              EFT1
3182
3183
3184                                    ;PUT ERROR TABLE HERE
3185                                    .SBTTL  ERROR TABLE USAGE
3186
3187                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3188                                    ;NUMBER, I.E.,
3189                                    ;
3190                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3191                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3192                                    ;       EDT - ERROR DATA TABLE ADDRESS
3193                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3194                                    ;
3195                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3196                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3197                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3198                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3199                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3200                    ;
3201                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3202                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3203                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3204                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3205                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3206                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3207                    ;MUST ALSO HAVE A FORMAT.
3208                    ;
3209                    ;IN SUMMARY,
3210                    ;
3211                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3212                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3213                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3214                    ;
3215
```

```
                                        .SBTTL  START OF PROGRAM

                                        START:

                                        ;CLEAR AND SETUP FOR TEST EXECUTION
                                                NOP
                                                RESET                   ;INITIALIZE THE SYSTEM
                                                MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
                                                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
                                                RTI                     ;;POP NEW PC AND PS
                                        64$:

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

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

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005612  005227  177777                          INC     #-1             ;;FIRST TIME?
005616  001061                                  BNE     69$             ;;BRANCH IF NO
005620  022737  033156  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
005626  001455                                  BEQ     69$             ;;BRANCH IF YES
005630  104401  005675                          TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005634  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005640  001012                                  BNE     71$             ;;BRANCH IF YES
005642  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005650  001406                                  BEQ     71$             ;;BRANCH IF YES
005652  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005660  001005                                  BNE     72$             ;;BRANCH IF NO
005662  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
005664  000403                                  BR      72$
005666  112737  000001  001150  71$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005674                                  72$:
005674  000432                                  BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@CZRM08 - RM03/RM02 SUBSYS FUNCTIONAL TEST,PART 2 @<CRLF>
005762                                  69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005762  122737  000077  000041                  CMPB    #77,@#41        ;LOAD FROM XXDP MEDIM ?
005770  001003                                  BNE     5$              ;BRANCH IF NOT
005772  104401  064562                          TYPE    ,XDPMG          ;TYPE THE MESSAGE
005776  000000                                  HALT
006000                                  5$:
006000  005737  000042                          TST     42              ;IS LOC 42 ZERO ??
006004  001003                                  BNE     10$             ;NO - NOT IN STANDALONE
006006  105737  001242                          TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
006012  001454                                  BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
006014                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
006014  132737  000200  001243  XSIZ:   BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
006022  001046                                  BNE     20$             ;NO
006024  012737  000377  001300                  MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
006032  005037  001300                          CLR     $DEVM           ;CLEAR THE BIT MAP
006036  005001                                  CLR     R1              ;START FROM THE DRIVE 0
006040  012704  000001                          MOV     #BIT0,R4        ;BIT MAP
006044  013700  001276                          MOV     $BASE,R0        ;LOAD BASE ADDRESS
006050  012760  000040  000010  15$:     MOV     #CLR,RMCS2(R0)  ;MASS BUS CLEAR
006056  010160  000010                          MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
006062  016003  000010                          MOV     RMCS2(R0),R3    ;READ DRIVE STATUS TO SEE NED BIT
006066  032703  010000                          BIT     #NED,R3         ;NED BIT SET ?
006072  001010                                  BNE     16$             ;BRANCH IF SO
006074  016003  000000                          MOV     RMCS1(R0),R3    ;CHECK THE DVA BIT
006100  032703  004000                          BIT     #DVA,R3         ;DRIVE AVAILABLE ?
006104  001403                                  BEQ     16$             ;BRANCH IF NOT
006106  050437  001300                          BIS     R4,$DEVM                ;SET THE BIT MAP
006112  000405                                  BR      17$             ;NOT TYPE THE NONE-EXIST MESG
006114                                  16$:
006114  104401  064647                          TYPE    ,NOTEX          ;NOT EXIST DRIVE
006120  010146                                  MOV     R1,-(SP)        ;DRIVE NUMBER
```

```
3328  006122  104403                        TYPOS
3329  006124     006                        .BYTE   6
3330  006125     000                        .BYTE   0
3331  006126  005201                  17$:  INC     R1              ;INCREMENT THE DRIVE ADDRESS
3332  006130  006304                        ASL     R4              ;SET UP BIT MAP
3333  006132  022701  000007                CMP     #7,R1           ;ALL DRIVE CHECKED ?
3334  006136  103344                        BHIS    15$             ;BRANCH IF NOT
3335  006140                            20$:
3336
3337                                  ;GO TO COMMON START CODE
3338  006140  000137  006772                JMP     CMNSTART
```

```
3339  006144                                  STANDALONE:
3340  006144  004737  06.400                          JSR     PC,$TKINT       ;INITIALIZE CONSOLE
3341
3342                                          ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3343  006150  005327  000001                        DEC     #1              ;FIRST START ??
3344  006154  100024                                BPL     10$             ;YES !!
3345
3346
3347                                          ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3348  006156  104401  064071                 5$:    TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
3349  006162  104411                                RDCHR                   ;GET RESPONSE
3350  006164  012637  001176                        MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3351  006170  104401  001176                        TYPE    ,$TMP1
3352  006174  123727  001176  000131                CMPB    $TMP1,#'Y       ;YES RESPONSE??
3353  006202  001002                                BNE     6$              ;NO!!
3354  006204  000137  007124                        JMP     READY           ;KEEP PREVIOUS PARAMETERS
3355  006210  123727  001176  000116          6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
3356  006216  001420                                BEQ     20$             ;GET NEW PARAMETERS
3357  006220  104401  063601                        TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
3358  006224  000754                                BR      5$              ;RETRY
3359  006226                                  10$:
3360
3361                                          ;SEE IF OPERATOR WANTS HELP FILE
3362  006226  104401  063603                        TYPE    ,HELPQST        ;WANT HELP ??
3363  006232  104411                                RDCHR                   ;GET RESPONSE
3364  006234  012637  001176                        MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3365  006240  104401  001176                        TYPE    ,$TMP1
3366  006244  123727  001176  000131                CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3367  006252  001002                                BNE     20$             ;NO - DONT TYPE HELP
3368  006254  104401  103566                        TYPE    ,HELP           ;YES - TYPE HELP TEXT
3369  006260                                  20$:
3370
3371                                          ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3372  006260  104401  063772                        TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
3373  006264  104411                                RDCHR                   ;GET RESPONSE
3374  006266  012637  001176                        MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3375  006272  104401  001176                        TYPE    ,$TMP1
3376  006276  104411                                RDCHR                   ;READ ONE MORE CHARACTER
3377  006300  005726                                TST     (SP)+           ;CLEAR OFF THE STACK
3378  006302  123727  001176  000131                CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3379  006310  001137                                BNE     90$             ;NO !!
3380  006312                                  30$:
3381
3382                                          ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3383  006312  104401  064130                        TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
3384  006316  013746  001276                        MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
3385  006322  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3386  006324  104401  063572                        TYPE    ,CLSPRN
3387  006330  104413                                RDOCT                   ;GET NEW BUS ADDRESS
3388  006332  012637  001176                        MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3389  006336  001412                                BEQ     50$             ;YES-SKIP TO NEXT ENTRY
3390  006340  022737  160000  001176                CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
3391  006346  101403                                BLOS    40$             ;YES
3392  006350  104401  064155                        TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
3393  006354  000756                                BR      30$             ;RETRY
3394  006356  013737  001176  001276          40$:  MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
```

```
3395  006364  113737  001272  001176  50$:    MOVB    $VECT1,$TMP1     ;TYPE CURRENT VECTOR ADDRESS
3396  006372  105037  001177                  CLRB    $TMP1+1
3397  006376  104401  064236                  TYPE    ,CNSL03
3398  006402  013746  001176                  MOV     $TMP1,-(SP)      ;;SAVE $TMP1 FOR TYPEOUT
3399  006406  104403                          TYPOS                    ;;GO TYPE--OCTAL ASCII
3400  006410     003                          .BYTE   3                ;;TYPE 3 DIGIT(S)
3401  006411     000                          .BYTE   0                ;;SUPPRESS LEADING ZEROS
3402  006412  104401  063572                  TYPE    ,CLSPRN
3403  006416  104413                          RDOCT                    ;GET NEW VECTOR ADDRESS
3404  006420  012637  001176                  MOV     (SP)+,$TMP1      ;CARRIAGE RETURN?
3405  006424  001412                          BEQ     70$              ;YES-SKIP TO NEXT ENTRY
3406  006426  022737  001000  001176          CMP     #1000,$TMP1      ;VECTOR ADDRESS < 1000??
3407  006434  101003                          BHI     60$              ;YES!!
3408  006436  104401  064266                  TYPE    ,CNSL04          ;TYPE WARNING MESSAGE
3409  006442  000750                          BR      50$              ;RETRY
3410  006444  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1     ;STORE NEW VECTOR ADDRESS
3411  006452  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1   ;TYPE CURRENT PRIORITY
3412  006460  006237  001176                  ASR     $TMP1
3413  006464  006237  001176                  ASR     $TMP1
3414  006470  006237  001176                  ASR     $TMP1
3415  006474  006237  001176                  ASR     $TMP1
3416  006500  006237  001176                  ASR     $TMP1
3417  006504  105037  001177                  CLRB    $TMP1+1
3418  006510  104401  064342                  TYPE    ,CNSL05
3419  006514  013746  001176                  MOV     $TMP1,-(SP)      ;;SAVE $TMP1 FOR TYPEOUT
3420  006520  104403                          TYPOS                    ;;GO TYPE--OCTAL ASCII
3421  006522     001                          .BYTE   1                ;;TYPE 1 DIGIT(S)
3422  006523     000                          .BYTE   0                ;;SUPPRESS LEADING ZEROS
3423  006524  104401  063572                  TYPE    ,CLSPRN
3424  006530  104413                          RDOCT                    ;GET NEW PRIORITY
3425  006532  012637  001176                  MOV     (SP)+,$TMP1      ;CARRIAGE RETURN??
3426  006536  001424                          BEQ     90$              ;YES-SKIP TO NEXT ENTRY
3427  006540  023727  001176  000007          CMP     $TMP1,#7         ;LEGAL PRIORITY??
3428  006546  002403                          BLT     80$              ;YES!!
3429  006550  104401  064376                  TYPE    ,CNSL06          ;TYPE WARNING MESSAGE
3430  006554  000736                          BR      70$              ;RETRY
3431  006556                          80$:                             ;STORE NEW PRIORITY
3432  006556  006337  001176                  ASL     $TMP1
3433  006562  006337  001176                  ASL     $TMP1
3434  006566  006337  001176                  ASL     $TMP1
3435  006572  006337  001176                  ASL     $TMP1
3436  006576  006337  001176                  ASL     $TMP1
3437  006602  113737  001176  001273          MOVB    $TMP1,$VECT1+1
3438  006610                          90$:
3439
3440                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3441  006610  005037  001300                  CLR     $DEVM            ;CLEAR DEVICE MAP
3442  006614  104401  064423                  TYPE    ,CNSL07          ;TYPE INPUT INSTRUCTIONS
3443  006620  104401  063575                  TYPE    ,PROMPT          ;TYPE PROMPTING CHARACTER
3444  006624  104411                          RDCHR                    ;GET RESPONSE
3445  006626  012637  001176                  MOV     (SP)+,$TMP1      ;ECHO RESPONSE
3446  006632  104401  001176                  TYPE    ,$TMP1
3447  006636  023727  001176  000101          CMP     $TMP1,#'A        ;TEST ALL DRIVES??
3448  006644  001021                          BNE     110$             ;NO
3449  006646  000137  006014                  JMP     XSIZ             ;AUTO SIZE
3450  006652  012737  000377  001300          MOV     #377,$DEVM       ;TEST ALL DEVICES
```

```
3451  006660  000444                          BR      140$            ;SKIP TO NEXT ENTRY
3452  006662  104401  063575          100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3453  006666  104411                          RDCHR                   ;GET RESPONSE
3454  006670  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3455  006674  104401  001176                  TYPE    ,$TMP1
3456  006700  023727  001176  000015          CMP     $TMP1,#CR       ;CARRIAGE RETURN??
3457  006706  001431                          BEQ     140$
3458  006710  023727  001176  000060  110$:   CMP     $TMP1,#'0       ;NUMBER < 0??
3459  006716  002404                          BLT     120$            ;YES!!
3460  006720  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7??
3461  006726  003403                          BLE     130$            ;NO!!
3462  006730  104401  063601          120$:   TYPE    ,QSTMRK         ;TYPE "?"
3463  006734  000752                          BR      100$            ;RETRY
3464  006736  013701  001176          130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
3465  006742  042701  177770                  BIC     #^C7,R1
3466  006746  116102  064774                  MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
3467  006752  042702  177400                  BIC     #^C377,R2               ;CLEAR UNUSED BITS
3468  006756  050237  001300                  BIS     R2,$DEVM                ;SET DEVICE # IN MAP
3469  006762  122737  000377  001300          CMPB    #377,$DEVM      ;DONE ??
3470  006770  101334                          BHI     100$            ;NO
3471  006772                          140$:
3472
```

```
3473  006772                          CMNSTART:
3474
3475                                  ;ASSEMBLE TEST QUE FROM DEVICE MAP
3476  006772  013700  001300                MOV     $DEVM,R0        ;R0 = DEVICE MAP
3477  006776  012701  001452                MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
3478  007002  010137  001450                MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
3479  007006  012702  000001                MOV     #1,R2           ;R2 = DEVICE POINTER
3480  007012  005003                        CLR     R3              ;R3 = DEVICE NUMBER
3481  007014  030200                10$:    BIT     R2,R0           ;IS THIS DEVICE IN MAP ??
3482  007016  001406                        BEQ     20$             ;NO !!
3483  007020  010311                        MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
3484  007022  116361  064774  000001        MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
3485  007030  062701  000002                ADD     #2,R1           ;ADVANCE ENTRY POINTER
3486  007034  006302                20$:    ASL     R2              ;ADVANCE DEVICE POINTER
3487  007036  105702                        TSTB    R2              ;DONE ALL DEVICES ??
3488  007040  001402                        BEQ     25$             ;YES
3489  007042  005203                        INC     R3              ;ADVANCE DEVICE NUMBER
3490  007044  000763                        BR      10$             ;ENTER NEXT DEVICE
3491  007046  005011                25$:    CLR     (R1)            ;TERMINATE TEST QUE
3492
3493                                  ;SIZE FOR CLOCK
3494  007050  004737  040204                JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
3495  007054  000403                        BR      40$             ;YES - CLOCK IS PRESENT
3496  007056  104000                30$:    ERROR                   ;NO CLOCK
3497  007060  000000                        HALT
3498  007062  000775                        BR      30$
3499  007064                          40$:
3500                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3501  007064  005737  000042                TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3502  007070  001012                        BNE     64$             ;;BRANCH IF YES
3503  007072  123727  001242  000001        CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
3504  007100  001406                        BEQ     64$             ;;BRANCH IF YES
3505  007102  023727  001154  000176        CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3506  007110  001005                        BNE     65$             ;;BRANCH IF NO
3507  007112  104407                        GTSWR                   ;;GET SOFT-SWR SETTINGS
3508  007114  000403                        BR      65$
3509  007116  112737  000001  001150  64$:  MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3510  007124                          65$:
3511  007124  000240                READY:  NOP                     ;READY TO START TEST
3512  007126  005037  001326                CLR     CTLFG           ;CLEAR THE CONTROL-C FLAG
3513  007132  004737  061400                JSR     PC,$TKINT       ;INITIALIZE TTY
3514  007136  013746  000300                MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
3515  007142  012746  007150                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3516  007146  000002                        RTI                     ;;POP NEW PC AND PS
3517  007150                          64$:
3518  007150  117737  172274  001234        MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
3519  007156  005037  001474                CLR     MEDENB          ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
007162                          TST1:
007162  000240                          NOP                             ;START OF TEST
007164  012737  007200  001122          MOV     #1$,$LPADR
007172  012737  007200  001124          MOV     #1$,$LPERR
007200                          1$:
007200  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007204  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007210  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007214  012737  000001  001226          MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
007222  005001                          CLR     R1
007224  013746  000004                  MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
007230  013746  000006                  MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
007234  012737  007326  000004          MOV     #3$,ERRVEC
007242  012737  000300  000006          MOV     #PR6,ERRVEC+2

007250  110160  000001                  MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
007254  010160  000002                  MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
007260  016002  000002                  MOV     RMWC(R0),R2
007264  010160  000004                  MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
007270  016002  000004                  MOV     RMBA(R0),R2
007274  010160  000010                  MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
007300  016002  000010                  MOV     RMCS2(R0),R2
007304  010160  000022                  MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
007310  016002  000022                  MOV     RMDB(R0),R2
007314  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007320  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007324  000415                          BR      7$                      ;NO BUS TIMEOUT OCCURRED

007326  022626                  3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
007330  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007334  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007340  104110                          ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
007342  005737  000042                  TST     42                      ;STAND ALONE MODE??
007346  001002                          BNE     5$                      ;NO!!
007350  000137  005324                  JMP     START                   ;YES-GO GET $BASE
007354  000137  032766          5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
007360                          7$:

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

                                ;;*****************************************************************
007360                          TST2:
007360  000004                          SCOPE                           ;SCOPE CALL
007362  000240                          NOP                             ;START OF TEST
007364  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007370  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007374  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007400  012737  000002  001226          MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

007406  004737  046660                  JSR     PC,CNTCLR
007412  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007414  000240                          NOP                             ;RETURN HERE IF ERROR
```

```
3576  007416  104000                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
3577  007420  000137  007540                JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
3578  007424                        2$:
3579  007424  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
3580  007430  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
3581  007434  012737  000004  000004        MOV     #5$,ERRVEC
3582  007442  013737  000030  000006        MOV     PR6,ERRVEC+2
3583
3584  007450  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
3585  007456  016037  000010  001174        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
3586  007464  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3587  007470  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3588  007474  032737  010000  001174        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
3589  007502  001402                        BEQ     3$              ;NO!!
3590  007504  104111                        ERROR   111             ;NONEXISTENT DEVICE
3591  007506  000414                        BR      7$
3592  007510  032737  004000  001176  3$:   BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
3593  007516  001012                        BNE     9$              ;YES!!
3594  007520  104112                        ERROR   112             ;DEVICE NOT AVAILABLE
3595  007522  000406                        BR      7$
3596
3597  007524  022626                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
3598  007526  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3599  007532  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3600  007536  104113                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
3601  007540  000137  032732        7$:     JMP     $EOSP
3602
3603  007544                        9$:
3604
3605                                ;;**************************************************************
3606                                ;*TEST 3         DRIVE TYPE TEST
3607
3608
3609  007544                        TST3:
3610  007544  000004                        SCOPE                           ;SCOPE CALL
3611  007546  000240                        NOP                             ;START OF TEST
3612  007550  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3613  007554  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3614  007560  013701  001450                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3615  007564  012737  000003  001226        MOV     #3,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3616
3617  007572  004737  046660                JSR     PC,CNTCLR
3618  007576  000404                        BR      2$      ;GO TO 2$ IF NO ERROR
3619  007600  000240                        NOP                             ;RETURN HERE IF ERROR
3620  007602  104000                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
3621  007604  000137  007722                JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
3622  007610                        2$:
3623  007610  112737  000026  001506        MOVB    #RMDT,GETINX            ;SETUP GET INDEX TABLE
3624  007616  112737  000200  001507        MOVB    #200,GETINX+1
3625  007624  012737  007726  001356        MOV     #5$,RMDTI               ;RMDT INPUT BUFFER = 5$
3626  007632  004737  037516                JSR     PC,GET                  ;GO GET DRIVE TYPE
3627  007636  000402                        BR      3$      ;GO TO 3$ IF NO ERROR
3628  007640  000240                        NOP                             ;RETURN HERE IF ERROR
3629  007642  104000                        ERROR                           ;ERROR NUMBER SPECIFIED BY GET
3630  007644  022737  020024  001356  3$:   CMP     #SNGPRT,RMDTI           ;SINGLE PORT RM03??
3631  007652  001425                        BEQ     5$                      ;YES!!
```

```
3632  007654  022737  024024  001356          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3633  007662  001421                          BEQ     5$              ;YES!!
3634  007664  022737  020025  001356          CMP     #SNGPRT!BIT0,RMDTI
3635  007672  001415                          BEQ     5$
3636  007674  022737  024025  001356          CMP     #DULPRT!BIT0,RMDTI
3637  007702  001411                          BEQ     5$
3638  007704  012737  020024  001176          MOV     #SNGPRT,$TMP1
3639  007712  012737  024024  001200          MOV     #DULPRT,$TMP2
3640  007720  104114                          ERROR   114             ;NOT AN RM03
3641  007722  000137  032732          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3642
3643  007726                          5$:
3644                                  ;;*****************************************************************
3645                                  ;*TEST 4         FORMAT ZEROS - 18
3646                                  ;;*****************************************************************
3647  007726                          TST4:
3648  007726  000004                          SCOPE                   ;SCOPE CALL
3649  007730  000240                          NOP                     ;START OF TEST
3650  007732  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3651  007736  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3652  007742  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3653  007746  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3654  007754                          10$:
3655
3656                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3657  007754  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
3658  007762  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
3659  007770  012737  000000  001432          MOV     #0,RMOF0                ;18 BIT FORMAT
3660  007776  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
3661  010004  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
3662  010012  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
3663  010020  012737  065106  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
3664  010026  012737  000001  001176          MOV     #1,$TMP1
3665  010034  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
3666  010040                          20$:
3667
3668                                  ;PREPARE DEVICE FOR DATA TRANSFER
3669  010040  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3670  010044  154130                          .WORD   154130  ;TASK DESCRIPTOR
3671  010046  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
3672  010050  000240                          NOP                     ;RETURN HERE IF ERROR
3673  010052  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3674  010054  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3675  010060                          30$:
3676
3677                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3678  010060  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
3679  010066  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
3680  010072  112722  000006                  MOVB    #RMDA,(R2)+
3681  010076  112722  000034                  MOVB    #RMDC,(R2)+
3682  010102  112722  000032                  MOVB    #RMOF,(R2)+
3683  010106  112722  000000                  MOVB    #RMCS1,(R2)+
3684  010112  112722  000200                  MOVB    #200,(R2)+
3685  010116  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3686  010122  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
3687  010124  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
3688  010126  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
3689  010130  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3690  010134                        40$:
3691
3692                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3693  010134  004737  037432                JSR     PC,GETSTS               ;SETUP FOR STATUS
3694  010140  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
3695  010144                        50$:
3696
3697                                ;GO READ SEEK STATUS
3698  010144  004737  037516                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
3699  010150  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
3700  010152  000240                        NOP                             ;RETURN HERE IF ERROR
3701  010154  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
3702  010156  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3703  010162                        60$:
3704
3705                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
3706  010162  004737  045420                JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
3707  010166  000405                        BR      70$                             ;GO TO 70$ IF NO ERROR
3708  010170  000240                        NOP                             ;RETURN HERE IF ERROR
3709  010172  104000                        ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
3710  010174  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3711  010176  000137  010526                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3712  010202                        70$:
3713
3714                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3715  010202  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
3716  010210  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
3717  010214  112723  000002                MOVB    #P**C,(R2)+
3718  010220  112722  000004                MOVB    #   A,(R2)+
3719  010224  112722  000000                MOVB    #RMCS1,(R2)+
3720  010230  112723  000200                MOVB    #200,(R2)+
3721  010234  004737  037766                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3722  010240  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
3723  010242  000240                        NOP                             ;RETURN HERE IF ERROR
3724  010244  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
3725  010246  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3726  010252                        80$:
3727
3728                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3729  010252  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
3730  010256  004737  037516                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
3731  010262  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
3732  010264  000240                        NOP                             ;RETURN HERE IF ERROR
3733  010266  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
3734  010270  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3735  010274                        90$:
3736
3737                                ;VERIFY RESULTS OF WRITE COMMAND
3738  010274  004737  040512                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
3739  010300  000405                        BR      100$                            ;GO TO 100$ IF NO ERROR
3740  010302  000240                        NOP                             ;RETURN HERE IF ERROR
3741  010304  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
3742  010306  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3743  010310  000137  010526                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
```

```
3744  010314                          100$:
3745  010314  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3746  010320  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
3747  010322  000240                          NOP                     ;RETURN HERE IF ERROR
3748  010324  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3749  010326  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3750  010330  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3751  010334                          110$:
3752  010334  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3753  010340  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
3754  010342  000240                          NOP                     ;RETURN HERE IF ERROR
3755  010344  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3756  010346  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3757  010350  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3758  010354                          120$:
3759
3760
3761                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3762  010354  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
3763  010362  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
3764  010370  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3765  010374  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
3766  010376  000240                          NOP                     ;RETURN HERE IF ERROR
3767  010400  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
3768  010402  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3769  010406                          130$:
3770
3771                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
3772  010406  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
3773  010412  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3774  010416  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
3775  010420  000240                          NOP                     ;RETURN HERE IF ERROR
3776  010422  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3777  010424  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3778  010430                          140$:
3779
3780                                  ;VERIFY THE RESULTS OF READ OPERATION
3781  010430  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3782  010434  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
3783  010436  000240                          NOP                     ;RETURN HERE IF ERROR
3784  010440  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3785  010442  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3786  010444  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3787  010450                          150$:
3788  010450  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3789  010454  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
3790  010456  000240                          NOP                     ;RETURN HERE IF ERROR
3791  010460  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3792  010462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3793  010464  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3794  010470                          160$:
3795  010470  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3796  010474  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
3797  010476  000240                          NOP                     ;RETURN HERE IF ERROR
3798  010500  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3799  010502  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
3800  010504  000137  010526                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3801  010510                            170$:
3802
3803                                    ;VERIFY DATA
3804  010510  004737  037064                    JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
3805  010514  103566                            .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
3806  010516  104572                            .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
3807  010520  000402                            BR      180$                            ;GO TO 180$ IF NO ERROR
3808  010522  000240                            NOP                             ;RETURN HERE IF ERROR
3809  010524  104000                            ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3810  010526                            180$:
3811
3812                                    ;;*****************************************************************
3813                                    ;*TEST 5         FORMAT ZEROS - 16
3814                                    ;;*****************************************************************
3815  010526                            TST5:
3816  010526  000004                            SCOPE                           ;SCOPE CALL
3817  010530  000240                            NOP                             ;START OF TEST
3818  010532  012706  001100                    MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3819  010536  013700  001276                    MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3820  010542  013701  001450                    MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3821  010546  012737  000005  001226            MOV     #5,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3822  010554                            10$:
3823
3824                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3825  010554  012737  000000  001434            MOV     #0,RMDC0                        ;CYLINDER = 0
3826  010562  012737  000000  001406            MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
3827  010570  012737  010000  001432            MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
3828  010576  012737  177376  001402            MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
3829  010604  012737  103566  001404            MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
3830  010612  012737  000062  001400            MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
3831  010620  012737  065106  001174            MOV     #ZEROS,$TMP0                    ;USE ALL ZEROS DATA PATTERN
3832  010626  012737  000001  001176            MOV     #1,$TMP1
3833  010634  004737  036620                    JSR     PC,GENBUF                       ;GO GENERATE DATA BUFFER
3834  010640                            20$:
3835
3836                                    ;PREPARE DEVICE FOR DATA TRANSFER
3837  010640  004737  034000                    JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
3838  010644  154130                            .WORD   154130  ;TASK DESCRIPTOR
3839  010646  000404                            BR      30$                             ;GO TO 30$ IF NO ERROR
3840  010650  000240                            NOP                             ;RETURN HERE IF ERROR
3841  010652  104000                            ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3842  010654  000137  011326                    JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3843  010660                            30$:
3844
3845                                    ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3846  010660  012737  000005  001400            MOV     #SEEK!GO,RMCS10                 ;CHANGE COMMAND TO SEEK
3847  010666  012702  001535                    MOV     #PUTINX,R2                      ;WRITE REGISTER INDEX TABLE
3848  010672  112722  000006                    MOVB    #RMDA,(R2)+
3849  010676  112722  000034                    MOVB    #RMDC,(R2)+
3850  010702  112722  000032                    MOVB    #RMOF,(R2)+
3851  010706  112722  000000                    MOVB    #RMCS1,(R2)+
3852  010712  112722  000200                    MOVB    #200,(R2)+
3853  010716  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3854  010722  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
3855  010724  000240                            NOP                             ;RETURN HERE IF ERROR
```

```
3856  010726  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
3857  010730  000137  011326                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3858  010734                        40$:
3859
3860                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3861  010734  004737  037432                JSR     PC,GETSTS               ;SETUP FOR STATUS
3862  010740  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
3863  010744                        50$:
3864
3865                                  ;GO READ SEEK STATUS
3866  010744  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3867  010750  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
3868  010752  000240                        NOP                     ;RETURN HERE IF ERROR
3869  010754  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
3870  010756  000137  011326                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3871  010762                        60$:
3872
3873                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
3874  010762  004737  045420                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
3875  010766  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
3876  010770  000240                        NOP                     ;RETURN HERE IF ERROR
3877  010772  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
3878  010774  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3879  010776  000137  011326                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3880  011002                        70$:
3881
3882                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3883  011002  012737  000063  001400        MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
3884  011010  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
3885  011014  112722  000002                MOVB    #RMWC,(R2)+
3886  011020  112722  000004                MOVB    #RMBA,(R2)+
3887  011024  112722  000000                MOVB    #RMCS1,(R2)+
3888  011030  112722  000200                MOVB    #200,(R2)+
3889  011034  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3890  011040  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
3891  011042  000240                        NOP                     ;RETURN HERE IF ERROR
3892  011044  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
3893  011046  000137  011326                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3894  011052                        80$:
3895
3896                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3897  011052  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
3898  011056  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3899  011062  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
3900  011064  000240                        NOP                     ;RETURN HERE IF ERROR
3901  011066  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
3902  011070  000137  011326                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3903  011074                        90$:
3904
3905                                  ;VERIFY RESULTS OF WRITE COMMAND
3906  011074  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3907  011100  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
3908  011102  000240                        NOP                     ;RETURN HERE IF ERROR
3909  011104  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3910  011106  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3911  011110  000137  011326                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
011114                          100$:
011114  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
011120  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
011122  000240                          NOP                     ;RETURN HERE IF ERROR
011124  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
011126  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011130  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
011134                          110$:
011134  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
011140  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
011142  000240                          NOP                     ;RETURN HERE IF ERROR
011144  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
011146  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011150  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
011154                          120$:


                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
011154  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
011162  012737  104572  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
011170  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
011174  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
011176  000240                          NOP                     ;RETURN HERE IF ERROR
011200  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
011202  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
011206                          130$:


                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
011206  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
011212  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
011216  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
011220  000240                          NOP                     ;RETURN HERE IF ERROR
011222  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
011224  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
011230                          140$:


                                ;VERIFY THE RESULTS OF READ OPERATION
011230  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
011234  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
011236  000240                          NOP                     ;RETURN HERE IF ERROR
011240  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
011242  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011244  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
011250                          150$:
011250  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
011254  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
011256  000240                          NOP                     ;RETURN HERE IF ERROR
011260  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
011262  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011264  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
011270                          160$:
011270  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
011274  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
011276  000240                          NOP                     ;RETURN HERE IF ERROR
011300  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
011302  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
 3968  011304  000137  011326                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 3969  011310                                170$:
 3970
 3971                                        ;VERIFY DATA
 3972  011310  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
 3973  011314  103566                        .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
 3974  011316  104572                        .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
 3975  011320  000402                        BR      180$                    ;GO TO 180$ IF NO ERROR
 3976  011322  000240                        NOP                     ;RETURN HERE IF ERROR
 3977  011324  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 3978  011326                                180$:
 3979
 3980                                        ;*****************************************************************
 3981                                        ;*TEST 6        ZERO FILL TEST
 3982                                        ;*****************************************************************
 3983  011326                                TST6:
 3984  011326  000004                        SCOPE                   ;SCOPE CALL
 3985  011330  000240                        NOP                     ;START OF TEST
 3986  011332  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 3987  011336  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 3988  011342  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 3989  011346  012737  000006  001226        MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 3990  011354                                10$:
 3991
 3992                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 3993  011354  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
 3994  011362  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 3995  011370  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 3996  011376  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 3997  011404  012737  103566  001404        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 3998  011412  012737  000062  001400        MOV     #WH.RMCS10              ;WRITE HEADER AND DATA
 3999  011420  012737  065106  001174        MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
 4000  011426  012737  000001  001176        MOV     #1,$TMP1
 4001  011434  004737  036620                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 4002  011440                                20$:
 4003
 4004                                        ;PREPARE DEVICE FOR DATA TRANSFER
 4005  011440  004737  034000                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 4006  011444  154130                        .WORD   ^154130  ;TASK DESCRIPTOR
 4007  011446  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
 4008  011450  000240                        NOP                     ;RETURN HERE IF ERROR
 4009  011452  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4010  011454  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4011  011460                                30$:
 4012
 4013                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 4014  011460  012737  000005  001400        MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
 4015  011466  012702  001535                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 4016  011472  112722  000006                MOVB    #RMDA,(R2)+
 4017  011476  112722  000034                MOVB    #RMDC,(R2)+
 4018  011502  112722  000032                MOVB    #RMOF,(R2)+
 4019  011506  112722  000000                MOVB    #RMCS1,(R2)+
 4020  011512  112722  000200                MOVB    #200,(R2)+
 4021  011516  004737  037766                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 4022  011522  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
 4023  011524  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
4024  011526  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4025  011530  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4026  011534                        40$:
4027
4028                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4029  011534  004737  037432                JSR     PC,GETSTS               ;SETUP FOR STATUS
4030  011540  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4031  011544                        50$:
4032
4033                                ;GO READ SEEK STATUS
4034  011544  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4035  011550  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
4036  011552  000240                        NOP                     ;RETURN HERE IF ERROR
4037  011554  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4038  011556  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4039  011562                        60$:
4040
4041                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
4042  011562  004737  045420                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4043  011566  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
4044  011570  000240                        NOP                     ;RETURN HERE IF ERROR
4045  011572  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4046  011574  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4047  011576  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4048  011602                        70$:
4049
4050                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4051  011602  012737  177776  001402        MOV     #<^C2+1),RMWC0          ;FORMAT PARTIAL SECTOR
4052  011610  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4053  011616  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4054  011622  112722  000002                MOVB    #RMWC,(R2)+
4055  011626  112722  000004                MOVB    #RMBA,(R2)+
4056  011632  112722  000000                MOVB    #RMCS1,(R2)+
4057  011636  112722  000200                MOVB    #200,(R2)+
4058  011642  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4059  011646  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
4060  011650  000240                        NOP                     ;RETURN HERE IF ERROR
4061  011652  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4062  011654  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4063  011660                        80$:
4064
4065                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4066  011660  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4067  011664  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4068  011670  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
4069  011672  000240                        NOP                     ;RETURN HERE IF ERROR
4070  011674  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4071  011676  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4072  011702                        90$:
4073
4074                                ;VERIFY RESULTS OF WRITE COMMAND
4075  011702  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4076  011706  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
4077  011710  000240                        NOP                     ;RETURN HERE IF ERROR
4078  011712  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4079  011714  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4080  011716  000137  012142            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4081  011722                          100$:
4082  011722  004737  053016            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4083  011726  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
4084  011730  000240                    NOP                     ;RETURN HERE IF ERROR
4085  011732  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4086  011734  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4087  011736  000137  012142            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4088  011742                          110$:
4089  011742  004737  041344            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4090  011746  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
4091  011750  000240                    NOP                     ;RETURN HERE IF ERROR
4092  011752  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4093  011754  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4094  011756  000137  012142            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4095  011762                          120$:
4096
4097
4098                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4099  011762  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWCO   ;READ A FULL SECTOR
4100  011770  012737  000073  001400        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
4101  011776  012737  104572  001404        MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
4102  012004  004737  037766            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4103  012010  000404                    BR      130$    ;GO TO 130$ IF NO ERROR
4104  012012  000240                    NOP                     ;RETURN HERE IF ERROR
4105  012014  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
4106  012016  000137  012142            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4107  012022                          130$:
4108
4109                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
4110  012022  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4111  012026  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4112  012032  000404                    BR      140$    ;GO TO 140$ IF NO ERROR
4113  012034  000240                    NOP                     ;RETURN HERE IF ERROR
4114  012036  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
4115  012040  000137  012142            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4116  012044                          140$:
4117
4118                                  ;VERIFY THE RESULTS OF READ OPERATION
4119  012044  004737  040512            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4120  012050  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
4121  012052  000240                    NOP                     ;RETURN HERE IF ERROR
4122  012054  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4123  012056  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4124  012060  000137  012142            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4125  012064                          150$:
4126  012064  004737  053016            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4127  012070  000405                    BR      160$                    ;GO TO 160$ IF NO ERROR
4128  012072  000240                    NOP                     ;RETURN HERE IF ERROR
4129  012074  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4130  012076  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4131  012100  000137  012142            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4132  012104                          160$:
4133  012104  004737  041344            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4134  012110  000405                    BR      170$                    ;GO TO 170$ IF NO ERROR
4135  012112  000240                    NOP                     ;RETURN HERE IF ERROR
```

```
 4136  012114  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
 4137  012116  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4138  012120  000137  012142                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
 4139  012124                          170$:
 4140
 4141                                  ;VERIFY DATA
 4142  012124  004737  037064                JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
 4143  012130  103566                        .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
 4144  012132  104572                        .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
 4145  012134  000402                        BR      180$                            ;GO TO 180$ IF NO ERROR
 4146  012136  000240                        NOP                             ;RETURN HERE IF ERROR
 4147  012140  104000                        ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 4148  012142                          180$:
 4149
 4150                                  ;;*****************************************************************
 4151                                  ;*TEST 7        FORMAT CHECK ZEROS - 16
 4152                                  ;;*****************************************************************
 4153  012142                          TST7:
 4154  012142  000004                        SCOPE                           ;SCOPE CALL
 4155  012144  000240                        NOP                             ;START OF TEST
 4156  012146  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
 4157  012152  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
 4158  012156  013701  001450                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
 4159  012162  012737  000007  001226        MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
 4160  012170                          10$:
 4161
 4162                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 4163  012170  012737  000000  001434        MOV     #0,RMDC0                        ;CYLINDER = 0
 4164  012176  012737  000000  001406        MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
 4165  012204  012737  010000  001432        MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
 4166  012212  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
 4167  012220  012737  103566  001404        MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
 4168  012226  012737  000062  001400        MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
 4169  012234  012737  065106  001174        MOV     #ZEROS,$TMP0                    ;USE ALL ZEROS DATA PATTERN
 4170  012242  012737  000001  001176        MOV     #1,$TMP1
 4171  012250  004737  036620                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 4172  012254                          20$:
 4173
 4174                                  ;PREPARE DEVICE FOR DATA TRANSFER
 4175  012254  004737  034000                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
 4176  012260  154130                        .WORD   154130  ;TASK DESCRIPTOR
 4177  012262  000404                        BR      30$                             ;GO TO 30$ IF NO ERROR
 4178  012264  000240                        NOP                             ;RETURN HERE IF ERROR
 4179  012266  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4180  012270  000137  012716                JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
 4181  012274                          30$:
 4182
 4183                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 4184  012274  012737  000005  001400        MOV     #SEEK!GO,RMCS10                 ;CHANGE COMMAND TO SEEK
 4185  012302  012702  001535                MOV     #PUTINX,R2                      ;WRITE REGISTER INDEX TABLE
 4186  012306  112722  000006                MOVB    #RMDA,(R2)+
 4187  012312  112722  000034                MOVB    #RMDC,(R2)+
 4188  012316  112722  000032                MOVB    #RMOF,(R2)+
 4189  012322  112722  000000                MOVB    #RMCS1,(R2)+
 4190  012326  112722  000200                MOVB    #200,(R2)+
 4191  012332  004737  037766                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
```

```
4192  012336  000404                    BR      40$     ;GO TO 40$ IF NO ERROR
4193  012340  000240                    NOP                     ;RETURN HERE IF ERROR
4194  012342  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
4195  012344  000137  012716            JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4196  012350                    40$:
4197
4198                            ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4199  012350  004737  037432            JSR     PC,GETSTS               ;SETUP FOR STATUS
4200  012354  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4201  012360                    50$:
4202
4203                            ;GO READ SEEK STATUS
4204  012360  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4205  012364  000404                    BR      60$     ;GO TO 60$ IF NO ERROR
4206  012366  000240                    NOP                     ;RETURN HERE IF ERROR
4207  012370  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
4208  012372  000137  012716            JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4209  012376                    60$:
4210
4211                            ;VERIFY THE RESULTS OF THE SEEK COMMAND
4212  012376  004737  045420            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4213  012402  000405                    BR      70$                     ;GO TO 70$ IF NO ERROR
4214  012404  000240                    NOP                     ;RETURN HERE IF ERROR
4215  012406  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4216  012410  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4217  012412  000137  012716            JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4218  012416                    70$:
4219
4220                            ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4221  012416  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4222  012424  012702  001540            MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4223  012430  112722  000002            MOVB    #RMWC,(R2)+
4224  012434  112722  000004            MOVB    #RMBA,(R2)+
4225  012440  112722  000000            MOVB    #RMCS1,(R2)+
4226  012444  112722  000200            MOVB    #200,(R2)+
4227  012450  004737  037766            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4228  012454  000404                    BR      80$     ;GO TO 80$ IF NO ERROR
4229  012456  000240                    NOP                     ;RETURN HERE IF ERROR
4230  012460  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
4231  012462  000137  012716            JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4232  012466                    80$:
4233
4234                            ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4235  012466  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4236  012472  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4237  012476  000404                    BR      90$     ;GO TO 90$ IF NO ERROR
4238  012500  000240                    NOP                     ;RETURN HERE IF ERROR
4239  012502  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
4240  012504  000137  012716            JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4241  012510                    90$:
4242
4243                            ;VERIFY RESULTS OF WRITE COMMAND
4244  012510  004737  040512            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4245  012514  000405                    BR      100$                    ;GO TO 100$ IF NO ERROR
4246  012516  000240                    NOP                     ;RETURN HERE IF ERROR
4247  012520  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
```

```
4248  012522  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4249  012524  000137  012716              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4250  012530                      100$:
4251  012530  004737  053016              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4252  012534  000405                      BR      110$                    ;GO TO 110$ IF NO ERROR
4253  012536  000240                      NOP                     ;RETURN HERE IF ERROR
4254  012540  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4255  012542  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4256  012544  000137  012716              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4257  012550                      110$:
4258  012550  004737  041344              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4259  012554  000405                      BR      120$                    ;GO TO 120$ IF NO ERROR
4260  012556  000240                      NOP                     ;RETURN HERE IF ERROR
4261  012560  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4262  012562  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4263  012564  000137  012716              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4264  012570                      120$:
4265
4266
4267                              ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4268  012570  012737  000053  001400      MOV     #WCH!GO,RMCS1           ;WRITE CHECK COMMAND
4269  012576  004737  037766              JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4270  012602  000404                      BR      130$    ;GO TO 130$ IF NO ERROR
4271  012604  000240                      NOP                     ;RETURN HERE IF ERROR
4272  012606  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
4273  012610  000137  012716              JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4274  012614                      130$:
4275
4276                              ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
4277  012614  004737  040326              JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4278  012620  004737  037516              JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4279  012624  000404                      BR      140$    ;GO TO 140$ IF NO ERROR
4280  012626  000240                      NOP                     ;RETURN HERE IF ERROR
4281  012630  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
4282  012632  000137  012716              JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4283  012636                      140$:
4284
4285                              ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4286  012636  004737  040512              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4287  012642  000405                      BR      150$                    ;GO TO 150$ IF NO ERROR
4288  012644  000240                      NOP                     ;RETURN HERE IF ERROR
4289  012646  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4290  012650  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4291  012652  000137  012716              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4292  012656                      150$:
4293  012656  004737  053016              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4294  012662  000405                      BR      160$                    ;GO TO 160$ IF NO ERROR
4295  012664  000240                      NOP                     ;RETURN HERE IF ERROR
4296  012666  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4297  012670  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4298  012672  000137  012716              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4299  012676                      160$:
4300  012676  004737  041344              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4301  012702  000405                      BR      170$                    ;GO TO 170$ IF NO ERROR
4302  012704  000240                      NOP                     ;RETURN HERE IF ERROR
4303  012706  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
```

```
4304  012710  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4305  012712  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4306  012716                          170$:
4307
4308  012716                          260$:
4309
4310                                  ;;*****************************************************************
4311                                  ;*TEST 10        FORMAT CHECK ZEROS W/ WCE ERROR
4312                                  ;;*****************************************************************
4313  012716                          TST10:
4314  012716  000004                          SCOPE                   ;SCOPE CALL
4315  012720  000240                          NOP                     ;START OF TEST
4316  012722  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4317  012726  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4318  012732  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BLING TESTED
4319  012736  012737  000010  001226          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4320  012744                          10$:
4321
4322                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4323  012744  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4324  012752  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
4325  012760  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4326  012766  012737  177376  001402          MOV     #<^C<2+25[illegible]>+1>,RMWC0  ;2 + 256 WORDS
4327  012774  012737  103566  001404          MOV     #BUFONE,R[illegible]A0  ;DATA BUFFER ADDRESS
4328  013002  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4329  013010  012737  065106  001174          MOV     #ZEROS,$TMP0            ;USE ALL ONES DATA PATTERN
4330  013016  012737  000001  001176          MOV     #1,$TMP1
4331  013024  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4332  013030                          20$:
4333
4334                                  ;PREPARE DEVICE FOR DATA TRANSFER
4335  013030  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4336  013034  154130                          .WORD   154130  ;TASK DESCRIPTOR
4337  013036  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
4338  013040  000240                          NOP                     ;RETURN HERE IF ERROR
4339  013042  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4340  013044  000137  013636                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4341  013050                          30$:
4342
4343                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4344  013050  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4345  013056  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4346  013062  112722  000006                  MOVB    #RMDA,(R2)+
4347  013066  112722  000034                  MOVB    #RMDC,(R2)+
4348  013072  112722  000032                  MOVB    #RMOF,(R2)+
4349  013076  112722  000000                  MOVB    #RMCS1,(R2)+
4350  013102  112722  000200                  MOVB    #200,(R2)+
4351  013106  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4352  013112  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4353  013114  000240                          NOP                     ;RETURN HERE IF ERROR
4354  013116  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4355  013120  000137  013636                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4356  013124                          40$:
4357
4358                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4359  013124  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
```

```
4360  013130  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4361  013134                        50$:
4362
4363                                ;GO READ SEEK STATUS
4364  013134  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4365  013140  000404                    BR      60$     ;GO TO 60$ IF NO ERROR
4366  013142  000240                    NOP                     ;RETURN HERE IF ERROR
4367  013144  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
4368  013146  000137  013636            JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4369  013152                        60$:
4370
4371                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
4372  013152  004737  045420            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4373  013156  000405                    BR      70$                     ;GO TO 70$ IF NO ERROR
4374  013160  000240                    NOP                     ;RETURN HERE IF ERROR
4375  013162  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4376  013164  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4377  013166  000137  013636            JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4378  013172                        70$:
4379
4380                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4381  013172  012737  000063  001400    MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
```

```
                                ;EXTEND REGISTER INDEX TABLE
013200  012702  001540          MOV     #PUTINX+3,R2
013204  112722  000002          MOVB    #RMWC,(R2)+
013210  112722  000004          MOVB    #RMBA,(R2)+
013214  112722  000000          MOVB    #RMCS1,(R2)+
013220  112722  000200          MOVB    #200,(R2)+
013224  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
013230  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
013232  000240                  NOP                     ;RETURN HERE IF ERROR
013234  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
013236  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
013242                  80$:

                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
013242  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
013246  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
013252  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
013254  000240                  NOP                     ;RETURN HERE IF ERROR
013256  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
013260  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
013264                  90$:

                        ;VERIFY RESULTS OF WRITE COMMAND
013264  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
013270  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
013272  000240                  NOP                     ;RETURN HERE IF ERROR
013274  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
013276  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
013300  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
013304                  100$:
013304  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
013310  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
013312  000240                  NOP                     ;RETURN HERE IF ERROR
013314  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
013316  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
013320  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
013324                  110$:
013324  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
013330  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
013332  000240                  NOP                     ;RETURN HERE IF ERROR
013334  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
013336  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
013340  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
013344                  120$:

                        ;ALTER DATA BUFFER
013344  005137  104570          COM     BUFTWO-2                ;COMPLEMENT LAST DATA WORD

                        ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
013350  012737  000053  001400  MOV     #WCH!GO,RMCS1           ;LOAD COMMAND
013356  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
013362  000404                  BR      180$    ;GO TO 180$ IF NO ERROR
013364  000240                  NOP                     ;RETURN HERE IF ERROR
013366  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
013370  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
013374                  180$:
```

```
4438
4439                                    ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
4440  013374  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMMAND TO COMPLETE
4441  013400  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4442  013404  000404                            BR      190$    ;GO TO 190$ IF NO ERROR
4443  013406  000240                            NOP                             ;RETURN HERE IF ERROR
4444  013410  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
4445  013412  000137  013636                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4446  013416                                190$:
4447
4448                                    ;CHECK FOR PRIMARY ERRORS
4449  013416  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4450  013422  000405                            BR      200$                    ;GO TO 200$ IF NO ERROR
4451  013424  000240                            NOP                     ;RETURN HERE IF ERROR
4452  013426  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4453  013430  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4454  013432  000137  013636                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4455  013436                                200$:
4456
```

```
                                        ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
013436  032737  040000  001340            BIT     #WCE,RMCS2I             ;IS WRITE CHECK ERROR SET??
013444  001023                            BNE     210$                    ;YES!!
013446  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
013452  000405                            BR      205$                    ;GO TO 205$ IF NO ERROR
013454  000240                            NOP                     ;RETURN HERE IF ERROR
013456  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
013460  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
013462  000137  013636                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
013466  013737  001340  001140  205$:     MOV     RMCS2I,$GDDAT           ;LOAD EXPECTED STATUS
013474  052737  040000  001140            BIS     #WCE,$GDDAT
013502  013737  001340  001142            MOV     RMCS2I,$BDDAT           ;LOAD RECEIVED STATUS
013510  104337                            ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
013512  000451                            BR      260$
013514                          210$:

                                        ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
013514  012737  104570  001134            MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
013522  013737  001334  001136            MOV     RMBAI,$BDADR            ;LOAD RECEIVED ADDRESS
013530  162737  000002  001136            SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS

                                        ;GET WCE DATA AND VERIFY IT IS OK
013536  112737  000022  001506            MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
013544  112737  000200  001507            MOVB    #200,GETINX+1
013552  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
013556  000404                            BR      230$    ;GO TO 230$ IF NO ERROR
013560  000240                            NOP                             ;RETURN HERE IF ERROR
013562  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
013564  000137  013636                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
013570  013737  001352  001142  230$:     MOV     RMDBI,$BDDAT            ;LOAD RECEIVED DATA WORD
013576  013737  104570  001140            MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
013604  005137  001140                    COM     $GDDAT
013610  023737  001134  001136            CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
013616  001402                            BEQ     220$                    ;YES!!
013620  104340                            ERROR   340                     ;ADDRESS OF WCE INCORRECT
013622  000405                            BR      260$
013624  023737  001140  001142  220$:     CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
013632  001401                            BEQ     260$                    ;YES!!
013634  104341                            ERROR   341                     ;UNEXPECTED WCE DATA
013636                          260$:

                                        ;;*****************************************************************
                                        ;*TEST 11       FORMAT ONES - 16
                                        ;;*****************************************************************
013636                          TST11:
013636  000004                            SCOPE                           ;SCOPE CALL
013640  000240                            NOP                             ;START OF TEST
013642  012706  001100                    MOV     #STACK,SP               ;INITIALIZE STACK POINTER
013646  013700  001276                    MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
013652  013701  001450                    MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
013656  012737  000011  001226            MOV     #11,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
013664                          10$:

                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
013664  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
013672  012737  000000  001406            MOV     #0,RMDA0                ;TRACK = SECTOR = 0
```

```
4513  013700  012737  010000  001432          MOV     #FMT16,RMOFO            ;16 BIT FORMAT
4514  013706  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWCO   ;2 + 256 WORDS
4515  013714  012737  103566  001404          MOV     #BUFONE,RMBAO           ;DATA BUFFER ADDRESS
4516  013722  012737  000062  001400          MOV     #WH,RMCS1O              ;WRITE HEADER AND DATA
4517  013730  012737  065044  001174          MOV     #ONES,$TMPJ             ;USE ALL ONES DATA PATTERN
4518  013736  012737  000001  001176          MOV     #1,$TMP1
4519  013744  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4520  013750                          20$:
4521
4522                                  ;PREPARE DEVICE FOR DATA TRANSFER
4523  013750  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4524  013754  154130                          .WORD   154130  ;TASK DESCRIPTOR
4525  013756  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
4526  013760  000240                          NOP                     ;RETURN HERE IF ERROR
4527  013762  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4528  013764  000137  014436                  JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
4529  013770                          30$:
4530
4531                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4532  013770  012737  000005  001400          MOV     #SEEK!GO,RMCS1O         ;CHANGE COMMAND TO SEEK
4533  013776  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4534  014002  112722  000006                  MOVB    #RMDA,(R2)+
4535  014006  112722  000034                  MOVB    #RMDC,(R2)+
4536  014012  112722  000032                  MOVB    #RMOF,(R2)+
4537  014016  112722  000000                  MOVB    #RMCS1,(R2)+
4538  014022  112722  000200                  MOVB    #200,(R2)+
4539  014026  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4540  014032  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4541  014034  000240                          NOP                     ;RETURN HERE IF ERROR
4542  014036  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4543  014040  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4544  014044                          40$:
4545
4546                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4547  014044  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
4548  014050  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4549  014054                          50$:
4550
4551                                  ;GO READ SEEK STATUS
4552  014054  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4553  014060  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
4554  014062  000240                          NOP                     ;RETURN HERE IF ERROR
4555  014064  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4556  014066  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4557  014072                          60$:
4558
4559                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
4560  014072  004737  045420                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4561  014076  000405                          BR      70$             ;GO TO 70$ IF NO ERROR
4562  014100  000240                          NOP                     ;RETURN HERE IF ERROR
4563  014102  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4564  014104  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4565  014106  000137  014436                  JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
4566  014112                          70$:
4567
4568                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4569  014112  012737  000063  001400          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4570  014120  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4571  014124  112722  000002                  MOVB    #RMWC,(R2)+
4572  014130  112722  000004                  MOVB    #RMBA,(R2)+
4573  014134  112722  000000                  MOVB    #RMCS1,(R2)+
4574  014140  112722  000200                  MOVB    #200,(R2)+
4575  014144  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4576  014150  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4577  014152  000240                          NOP                             ;RETURN HERE IF ERROR
4578  014154  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4579  014156  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4580  014162                          80$:
4581
4582                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4583  014162  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4584  014166  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4585  014172  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4586  014174  000240                          NOP                             ;RETURN HERE IF ERROR
4587  014176  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4588  014200  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4589  014204                          90$:
4590
4591                                  ;VERIFY RESULTS OF WRITE COMMAND
4592  014204  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
4593  014210  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4594  014212  000240                          NOP                             ;RETURN HERE IF ERROR
4595  014214  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4596  014216  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4597  014220  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4598  014224                          100$:
4599  014224  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4600  014230  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4601  014232  000240                          NOP                             ;RETURN HERE IF ERROR
4602  014234  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4603  014236  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4604  014240  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4605  014244                          110$:
4606  014244  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4607  014250  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4608  014252  000240                          NOP                             ;RETURN HERE IF ERROR
4609  014254  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4610  014256  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4611  014260  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4612  014264                          120$:
4613
4614
4615                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4616  014264  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
4617  014272  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
4618  014300  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4619  014304  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4620  014306  000240                          NOP                             ;RETURN HERE IF ERROR
4621  014310  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4622  014312  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4623  014316                          130$:
4624
```

```
                                        ;WAIT FOR READ TO COMPLETE AND GET STATUS
014316  004737  040326                          JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
014322  004737  037516                          JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
014326  000404                                  BR      140$    ;GO TO 140$ IF NO ERROR
014330  000240                                  NOP                             ;RETURN HERE IF ERROR
014332  104000                                  ERROR                           ;ERROR DEFINED BY GET SUB
014334  000137  014436                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
014340                                  140$:

                                        ;VERIFY THE RESULTS OF READ OPERATION
014340  004737  040512                          JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
014344  000405                                  BR      150$                            ;GO TO 150$ IF NO ERROR
014346  000240                                  NOP                             ;RETURN HERE IF ERROR
014350  104000                                  ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
014352  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
014354  000137  014436                          JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
014360                                  150$:
014360  004737  053016                          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
014364  000405                                  BR      160$                            ;GO TO 160$ IF NO ERROR
014366  000240                                  NOP                             ;RETURN HERE IF ERROR
014370  104000                                  ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
014372  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
014374  000137  014436                          JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
014400                                  160$:
014400  004737  041344                          JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
014404  000405                                  BR      170$                            ;GO TO 170$ IF NO ERROR
014406  000240                                  NOP                             ;RETURN HERE IF ERROR
014410  104000                                  ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
014412  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
014414  000137  014436                          JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
014420                                  170$:

                                        ;VERIFY DATA
014420  004737  037064                          JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
014424  103566                                  .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
014426  104572                                  .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
014430  000402                                  BR      180$                            ;GO TO 180$ IF NO ERROR
014432  000240                                  NOP                             ;RETURN HERE IF ERROR
014434  104000                                  ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
014436                                  180$:

                                        ;;*****************************************************************
                                        ;*TEST 12       FORMAT CHECK ONES - 16
                                        ;;*****************************************************************
014436                                  TST12:
014436  000004                                  SCOPE                           ;SCOPE CALL
014440  000240                                  NOP                             ;START OF TEST
014442  012706  001100                          MOV     #STACK,SP               ;INITIALIZE STACK POINTER
014446  013700  001276                          MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
014452  013701  001450                          MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
014456  012737  000012  001226                  MOV     #12,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
014464                                  10$:

                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
014464  012737  000000  001434                  MOV     #0,RMDC0                        ;CYLINDER = 0
014472  012737  000000  001406                  MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
```

```
4681  014500  012737  010000  001432          MOV     #FMT16,RMOFO            ;16 BIT FORMAT
4682  014506  012737  177376  001402          MOV     #<^C<2^256.>+1>,RMWCO   ;2 + 256 WORDS
4683  014514  012737  103566  001404          MOV     #BUFONE,RMBAO           ;DATA BUFFER ADDRESS
4684  014522  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4685  014530  012737  065044  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
4686  014536  012737  000001  001176          MOV     #1,$TMP1
4687  014544  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4688  014550                                20$:
4689
4690                                        ;PREPARE DEVICE FOR DATA TRANSFER
4691  014550  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4692  014554  154130                          .WORD   154130  ;TASK DESCRIPTOR
4693  014556  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
4694  014560  000240                          NOP                     ;RETURN HERE IF ERROR
4695  014562  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4696  014564  000137  015212                  JMP     260$            ;GO TO 260$ IF ERROR WAS FOUND
4697  014570                                30$:
4698
4699                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4700  014570  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4701  014576  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4702  014602  112722  000006                  MOVB    #RMDA,(R2)+
4703  014606  112722  000034                  MOVB    #RMDC,(R2)+
4704  014612  112722  000032                  MOVB    #RMOF,(R2)+
4705  014616  112722  000000                  MOVB    #RMCS1,(R2)+
4706  014622  112722  000200                  MOVB    #200,(R2)+
4707  014626  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4708  014632  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4709  014634  000240                          NOP                     ;RETURN HERE IF ERROR
4710  014636  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4711  014640  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4712  014644                                40$:
4713
4714                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEFK TO COMPLETE
4715  014644  004737  037432                  JSR     PC,GETSTS               ;SETUF FOR STATUS
4716  014650  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4717  014654                                50$:
4718
4719                                        ;GO READ SEEK STATUS
4720  014654  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4721  014660  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
4722  014662  000240                          NOP                     ;RETURN HERE IF ERROR
4723  014664  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4724  014666  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4725  014672                                60$:
4726
4727                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
4728  014672  004737  045420                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4729  014676  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
4730  014700  000240                          NOP                     ;RETURN HERE IF ERROR
4731  014702  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4732  014704  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4733  014706  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4734  014712                                70$:
4735
4736                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4737  014712  012737  000063  001400          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4738  014720  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4739  014724  112722  000002                  MOVB    #RMWC,(R2)+
4740  014730  112722  000004                  MOVB    #RMBA,(R2)+
4741  014734  112722  000000                  MOVB    #RMCS1,(R2)+
4742  014740  112722  000200                  MOVB    #200,(R2)+
4743  014744  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4744  014750  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4745  014752  000240                          NOP                             ;RETURN HERE IF ERROR
4746  014754  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4747  014756  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4748  014762                          80$:
4749
4750                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4751  014762  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4752  014766  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4753  014772  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4754  014774  000240                          NOP                             ;RETURN HERE IF ERROR
4755  014776  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4756  015000  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4757  015004                          90$:
4758
4759                                  ;VERIFY RESULTS OF WRITE COMMAND
4760  015004  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
4761  015010  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4762  015012  000240                          NOP                             ;RETURN HERE IF ERROR
4763  015014  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4764  015016  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4765  015020  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4766  015024                          100$:
4767  015024  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4768  015030  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4769  015032  000240                          NOP                             ;RETURN HERE IF ERROR
4770  015034  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4771  015036  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4772  015040  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4773  015044                          110$:
4774  015044  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4775  015050  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4776  015052  000240                          NOP                             ;RETURN HERE IF ERROR
4777  015054  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4778  015056  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4779  015060  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4780  015064                          120$:
4781
4782
4783                                  ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4784  015064  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;WRITE CHECK COMMAND
4785  015072  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4786  015076  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4787  015100  000240                          NOP                             ;RETURN HERE IF ERROR
4788  015102  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4789  015104  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4790  015110                          130$:
4791
4792                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
```

```
 4793  015110  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
 4794  015114  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4795  015120  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
 4796  015122  000240                          NOP                     ;RETURN HERE IF ERROR
 4797  015124  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 4798  015126  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4799  015132                          140$:
 4800
 4801                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
 4802  015132  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
 4803  015136  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
 4804  015140  000240                          NOP                     ;RETURN HERE IF ERROR
 4805  015142  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 4806  015144  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4807  015146  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4808  015152                          150$:
 4809  015152  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
 4810  015156  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
 4811  015160  000240                          NOP                     ;RETURN HERE IF ERROR
 4812  015162  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4813  015164  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4814  015166  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4815  015172                          160$:
 4816  015172  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
 4817  015176  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
 4818  015200  000240                          NOP                     ;RETURN HERE IF ERROR
 4819  015202  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 4820  015204  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4821  015206  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4822  015212                          170$:
 4823
 4824  015212                          260$:
 4825
 4826                                  ;;*****************************************************************
 4827                                  ;*TEST 13       FORMAT CHECK ONES W/ WCE ERRORS
 4828                                  ;;*****************************************************************
 4829  015212                          TST13:
 4830  015212  000004                          SCOPE                           ;SCOPE CALL
 4831  015214  000240                          NOP                             ;START OF TEST
 4832  015216  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
 4833  015222  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
 4834  015226  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
 4835  015232  012737  000013  001226          MOV     #13,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
 4836  015240                          10$:
 4837
 4838                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 4839  015240  012737  000000  001434          MOV     #0,RMDC0                        ;CYLINDER = 0
 4840  015246  012737  000000  001406          MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
 4841  015254  012737  010000  001432          MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
 4842  015262  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
 4843  015270  012737  103566  001404          MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
 4844  015276  012737  000062  001400          MOV     #WH,RMCS1O                      ;WRITE HEADER AND DATA
 4845  015304  012737  065044  001174          MOV     #ONES,$TMP0                     ;USE ALL ONES DATA PATTERN
 4846  015312  012737  000001  001176          MOV     #1,$TMP1
 4847  015320  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 4848  015324                          20$:
```

```
                                        ;PREPARE DEVICE FOR DATA TRANSFER
015324  004737  034000                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
015330  154130                                  .WORD   154130  ;TASK DESCRIPTOR
015332  000404                                  BR      30$                     ;GO TO 30$ IF NO ERROR
015334  000240                                  NOP                     ;RETURN HERE IF ERROR
015336  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015340  000137  016130                          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
015344                                  30$:

                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
015344  012737  000005  001400                  MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
015352  012702  001535                          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
015356  112722  000006                          MOVB    #RMDA,(R2)+
015362  112722  000034                          MOVB    #RMDC,(R2)+
015366  112722  000032                          MOVB    #RMOF,(R2)+
015372  112722  000000                          MOVB    #RMCS1,(R2)+
015376  112722  000200                          MOVB    #200,(R2)+
015402  004737  037766                          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
015406  000404                                  BR      40$     ;GO TO 40$ IF NO ERROR
015410  000240                                  NOP                     ;RETURN HERE IF ERROR
015412  104000                                  ERROR                   ;ERROR DEFINED BY PUT SUB
015414  000137  016130                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
015420                                  40$:

                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
015420  004737  037432                          JSR     PC,GETSTS               ;SETUP FOR STATUS
015424  004737  040326                          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
015430                                  50$:

                                        ;GO READ SEEK STATUS
015430  004737  037516                          JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
015434  000404                                  BR      60$     ;GO TO 60$ IF NO ERROR
015436  000240                                  NOP                     ;RETURN HERE IF ERROR
015440  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
015442  000137  016130                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
015446                                  60$:

                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
015446  004737  045420                          JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
015452  000405                                  BR      70$                     ;GO TO 70$ IF NO ERROR
015454  000240                                  NOP                     ;RETURN HERE IF ERROR
015456  104000                                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
015460  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
015462  000137  016130                          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
015466                                  70$:

                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
015466  012737  000063  001400                  MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
015474  012702  001540                          MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
015500  112722  000002                          MOVB    #RMWC,(R2)+
015504  112722  000004                          MOVB    #RMBA,(R2)+
015510  112722  000000                          MOVB    #RMCS1,(R2)+
015514  112722  000200                          MOVB    #200,(R2)+
015520  004737  037766                          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
015524  000404                                  BR      80$     ;GO TO 80$ IF NO ERROR
```

```
4905  015526  000240                          NOP                     ;RETURN HERE IF ERROR
4906  015530  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4907  015532  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4908  015536                          80$:
4909
4910                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4911  015536  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4912  015542  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4913  015546  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4914  015550  000240                          NOP                     ;RETURN HERE IF ERROR
4915  015552  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4916  015554  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4917  015560                          90$:
4918
4919                                  ;VERIFY RESULTS OF WRITE COMMAND
4920  015560  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4921  015564  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4922  015566  000240                          NOP                     ;RETURN HERE IF ERROR
4923  015570  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4924  015572  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4925  015574  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4926  015600                          100$:
4927  015600  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4928  015604  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4929  015606  000240                          NOP                     ;RETURN HERE IF ERROR
4930  015610  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4931  015612  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4932  015614  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4933  015620                          110$:
4934  015620  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4935  015624  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4936  015626  000240                          NOP                     ;RETURN HERE IF ERROR
4937  015630  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4938  015632  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4939  015634  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4940  015640                          120$:
4941
4942                                  ;ALTER DATA BUFFER
4943  015640  005137  104570                  COM     BUFTWO-2                ;COMPLEMENT DATA WORD
4944
4945
4946                                  ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
4947  015644  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;LOAD COMMAND
4948  015652  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4949  015656  000404                          BR      180$    ;GO TO 180$ IF NO ERROR
4950  015660  000240                          NOP                     ;RETURN HERE IF ERROR
4951  015662  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4952  015664  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4953  015670                          180$:
4954
4955                                  ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
4956  015670  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMMAND TO COMPLETE
4957  015674  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4958  015700  000404                          BR      190$    ;GO TO 190$ IF NO ERROR
4959  015702  000240                          NOP                     ;RETURN HERE IF ERROR
4960  015704  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
```

```
4961  015706  000137  016130              JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4962  015712                      190$:
4963
4964                              ;CHECK FOR PRIMARY ERRORS
4965  015712  004737  040512              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4966  015716  000405                      BR      200$                    ;GO TO 200$ IF NO ERROR
4967  015720  000240                      NOP                     ;RETURN HERE IF ERROR
4968  015722  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4969  015724  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4970  015726  000137  016130              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4971  015732                      200$:
4972
4973                              ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4974  015732  032737  040000  001340      BIT     #WCE,RMCS2I             ;IS WRITE CHECK ERROR SET??
4975  015740  001022                      BNE     210$                    ;YES!!
4976  015742  004737  053016              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4977  015746  000405                      BR      205$                    ;GO TO 205$ IF NO ERROR
4978  015750  000240                      NOP                     ;RETURN HERE IF ERROR
4979  015752  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4980  015754  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4981  015756  000137  016130              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4982  015762  013737  001340  001140  205$:  MOV     RMCS2I,$GDDAT           ;LOAD EXPECTED STATUS
4983  015770  052737  040000  001140      BIS     #WCE,$GDDAT
4984  015776  013737  001340  001142      MOV     RMCS2I,$BDDAT           ;LOAD RECEIVED STATUS
4985  016004  104337                      ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
4986  016006                      210$:
4987
4988                              ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4989  016006  012737  104570  001134      MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
4990  016014  013737  001334  001136      MOV     R0SAI,$BDADR            ;LOAD RECEIVED ADDRESS
4991  016022  162737  000002  001136      SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS
4992
4993                              ;GET WCE DATA AND VERIFY IT IS OK
4994  016030  112737  000022  001506      MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
4995  016036  112737  000200  001507      MOVB    #200,GETINX+1
4996  016044  004737  037516              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4997  016050  000404                      BR      230$    ;GO TO 230$ IF NO ERROR
4998  016052  000240                      NOP                     ;RETURN HERE IF ERROR
4999  016054  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
5000  016056  000137  016130              JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
5001  016062  013737  001352  001142  230$:  MOV     RMDBI,$BDDAT            ;LOAD RECEIVED DATA WORD
5002  016070  013737  104570  001140      MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
5003  016076  005137  001140              COM     $GDDAT
5004  016102  023737  001134  001136      CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
5005  016110  001402                      BEQ     220$                    ;YES!!
5006  016112  104340                      ERROR   340                     ;ADDRESS OF WCE INCORRECT
5007  016114  000405                      BR      260$
5008  016116                      220$:
5009  016116  023737  001140  001142      CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
5010  016124  001401                      BEQ     260$                    ;YES!!
5011  016126  104341                      ERROR   341                     ;UNEXPECTED WCE DATA
5012  016130                      260$:
5013
5014                              ;;*****************************************************************
5015                              ;*TEST 14        FORMAT MULTIPLE SECTORS
5016                              ;;*****************************************************************
```

```
                        TST14:
016130                          SCOPE                   ;SCOPE CALL
016130  000004                  NOP                     ;START OF TEST
016132  000240                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
016134  012706  001100          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
016140  013700  001276          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
016144  013701  001450          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016150  012737  000014  001226
                        10$:
016156
                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
016156  012737  000000  001434  MOV     #0,RMDC0                ;CYLINDER = 0
016164  012737  000000  001406  MOV     #0,RMDA0                ;TRACK = SECTOR = 0
016172  012737  010000  001432  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
016200  012737  176774  001402  MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
016206  012737  103566  001404  MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
016214  012737  000062  001400  MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
016222  012737  065106  001174  MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
016230  012737  000001  001176  MOV     #1,$TMP1
016236  004737  036620          JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
016242                  20$:
                        ;PREPARE DEVICE FOR DATA TRANSFER
016242  004737  034000          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
016246  154130                  .WORD   154130  ;TASK DESCRIPTOR
016250  000404                  BR      30$             ;GO TO 30$ IF NO ERROR
016252  000240                  NOP                     ;RETURN HERE IF ERROR
016254  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
016256  000137  016704          JMP     18O$            ;GO TO 18O$ IF ERROR WAS FOUND
016262                  30$:
                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
016262  012737  000005  001400  MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
016270  012702  001535          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
016274  112722  000006          MOVB    #RMDA,(R2)+
016300  112722  000034          MOVB    #RMDC,(R2)+
016304  112722  000032          MOVB    #RMOF,(R2)+
016310  112722  000000          MOVB    #RMCS1,(R2)+
016314  112722  000200          MOVB    #200,(R2)+
016320  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
016324  000404                  BR      40$     ;GO TO 40$ IF NO ERROR
016326  000240                  NOP                     ;RETURN HERE IF ERROR
016330  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
016332  000137  016704          JMP     18O$    ;GO TO 18O$ IF ERROR WAS FOUND
016336                  40$:
                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
016336  004737  037432          JSR     PC,GETSTS               ;SETUP FOR STATUS
016342  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
016346                  50$:
                        ;GO READ SEEK STATUS
016346  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
016352  000404                  BR      60$     ;GO TO 60$ IF NO ERROR
016354  000240                  NOP                     ;RETURN HERE IF ERROR
016356  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
016360  000137  016704          JMP     18O$    ;GO TO 18O$ IF ERROR WAS FOUND
```

```
5073  016364                                    60$:
5074
5075                                            ;VERIFY THE RESULTS OF THE SEEK COMMAND
5076  016364  004737  045420                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5077  016370  000405                            BR      70$                     ;GO TO 70$ IF NO ERROR
5078  016372  000240                            NOP                     ;RETURN HERE IF ERROR
5079  016374  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5080  016376  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5081  016400  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5082  016404                                    70$:
5083
5084                                            ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5085  016404  012737  000063  001400            MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
5086  016412  012702  001540                    MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5087  016416  112722  000002                    MOVB    #RMWC,(R2)+
5088  016422  112722  000004                    MOVB    #RMBA,(R2)+
5089  016426  112722  000000                    MOVB    #RMCS1,(R2)+
5090  016432  112722  000200                    MOVB    #200,(R2)+
5091  016436  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5092  016442  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5093  016444  000240                            NOP                     ;RETURN HERE IF ERROR
5094  016446  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5095  016450  000137  016704                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5096  016454                                    80$:
5097
5098                                            ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5099  016454  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5100  016460  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5101  016464  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5102  016466  000240                            NOP                     ;RETURN HERE IF ERROR
5103  016470  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5104  016472  000137  016704                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5105  016476                                    90$:
5106
5107                                            ;VERIFY RESULTS OF WRITE COMMAND
5108  016476  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5109  016502  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
5110  016504  000240                            NOP                     ;RETURN HERE IF ERROR
5111  016506  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5112  016510  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5113  016512  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5114  016516                                    100$:
5115  016516  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5116  016522  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
5117  016524  000240                            NOP                     ;RETURN HERE IF ERROR
5118  016526  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5119  016530  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5120  016532  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5121  016536                                    110$:
5122  016536  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5123  016542  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
5124  016544  000240                            NOP                     ;RETURN HERE IF ERROR
5125  016546  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5126  016550  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5127  016552  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5128  016556                                    120$:
```

```
                                        ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
016556  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;READ HEADER & DATA COMMAND
016564  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
016570  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
016572  000240                          NOP                             ;RETURN HERE IF ERROR
016574  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
016576  000137  016704                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
016602                          130$:
                                        ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
016602  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
016606  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
016612  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
016614  000240                          NOP                             ;RETURN HERE IF ERROR
016616  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
016620  000137  016704                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
016624                          140$:
                                        ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
016624  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
016630  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
016632  000240                          NOP                             ;RETURN HERE IF ERROR
016634  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
016636  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016640  000137  016704                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
016644                          150$:
016644  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
016650  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
016652  000240                          NOP                             ;RETURN HERE IF   ROR
016654  104000                          ERROR                           ;ERROR # DEFINE, BY DTASTS SUBROUTINE
016656  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016660  000137  016704                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
016664                          160$:
016664  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
016670  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
016672  000240                          NOP                             ;RETURN HERE IF ERROR
016674  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
016676  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016700  000137  016704                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
016704                          170$:
016704                          180$:

                                        ;;*****************************************************************
                                        ;*TEST 15        FORMAT W/ HEAD SWITCHING
                                        ;;*****************************************************************
016704                          TST15:
016704  000004                          SCOPE                           ;SCOPE CALL
016706  000240                          NOP                             ;START OF TEST
016710  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
016714  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
016720  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
016724  012737  000015  001226          MOV     #15,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
016732                          10$:
```

```
5185                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5186  016732  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
5187  016740  012737  000037  001406            MOV     #31.,RMDA0              ;START AT LAST SECTOR
5188  016746  012737  010000  001432            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5189  016754  012737  176774  001402            MOV     #^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5190  016762  012737  103566  001404            MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5191  016770  012737  000062  001400            MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5192  016776  012737  065106  001174            MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5193  017004  012737  000001  001176            MOV     #1,$TMP1
5194  017012  004737  036620                    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5195  017016                                20$:
5196
5197                                    ;PREPARE DEVICE FOR DATA TRANSFER
5198  017016  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5199  017022  154130                            .WORD   154130  ;TASK DESCRIPTOR
5200  017024  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
5201  017026  000240                            NOP                     ;RETURN HERE IF ERROR
5202  017030  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5203  017032  000137  017460                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5204  017036                                30$:
5205
5206                                    ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5207  017036  012737  000005  001400            MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5208  017044  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5209  017050  112722  000006                    MOVB    #RMDA,(R2)+
5210  017054  112722  000034                    MOVB    #RMDC,(R2)+
5211  017060  112722  000032                    MOVB    #RMOF,(R2)+
5212  017064  112722  000000                    MOVB    #RMCS1,(R2)+
5213  017070  112722  000200                    MOVB    #200,(R2)+
5214  017074  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5215  017100  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
5216  017102  000240                            NOP                     ;RETURN HERE IF ERROR
5217  017104  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5218  017106  000137  017460                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5219  017112                                40$:
5220
5221                                    ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5222  017112  004737  037432                    JSR     PC,GETSTS               ;SETUP FOR STATUS
5223  017116  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5224  017122                                50$:
5225
5226                                    ;GO READ SEEK STATUS
5227  017122  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5228  017126  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
5229  017130  000240                            NOP                     ;RETURN HERE IF ERROR
5230  017132  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5231  017134  000137  017460                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5232  017140                                60$:
5233
5234                                    ;VERIFY THE RESULTS OF THE SEEK COMMAND
5235  017140  004737  045420                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5236  017144  000405                            BR      70$                     ;GO TO 70$ IF NO ERROR
5237  017146  000240                            NOP                     ;RETURN HERE IF ERROR
5238  017150  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5239  017152  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5240  017154  000137  017460                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
                                        70$:

                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
017160  012737  000063  001400                MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
017166  012702  001540                        MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
017172  112722  000002                        MOVB    #RMWC,(R2)+
017176  112722  000004                        MOVB    #RMBA,(R2)+
017202  112722  000000                        MOVB    #RMCS1,(R2)+
017206  112722  000200                        MOVB    #200,(R2)+
017212  004737  037766                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
017216  000404                                BR      80$     ;GO TO 80$ IF NO ERROR
017220  000240                                NOP                             ;RETURN HERE IF ERROR
017222  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
017224  000137  017460                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
017230                                  80$:

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
017230  004737  040326                        JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
017234  004737  037516                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
017240  000404                                BR      90$     ;GO TO 90$ IF NO ERROR
017242  000240                                NOP                             ;RETURN HERE IF ERROR
017244  104000                                ERROR                           ;ERROR DEFINED BY GET SUB
017246  000137  017460                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
017252                                  90$:

                                        ;VERIFY RESULTS OF WRITE COMMAND
017252  004737  040512                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
017256  000405                                BR      100$                            ;GO TO 100$ IF NO ERROR
017260  000240                                NOP                             ;RETURN HERE IF ERROR
017262  104000                                ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
017264  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
017266  000137  017460                        JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
017272                                  100$:
017272  004737  053016                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
017276  000405                                BR      110$                            ;GO TO 110$ IF NO ERROR
017300  000240                                NOP                             ;RETURN HERE IF ERROR
017302  104000                                ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
017304  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
017306  000137  017460                        JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
017312                                  110$:
017312  004737  041344                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
017316  000405                                BR      120$                            ;GO TO 120$ IF NO ERROR
017320  000240                                NOP                             ;RETURN HERE IF ERROR
017322  104000                                ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
017324  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
017326  000137  017460                        JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
017332                                  120$:

                                        ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
017332  012737  000053  001400                MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
017340  004737  037766                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
017344  000404                                BR      130$    ;GO TO 130$ IF NO ERROR
017346  000240                                NOP                             ;RETURN HERE IF ERROR
017350  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
017352  000137  017460                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5297  017356                                  130$:
5298
5299                                          ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5300  017356  004737  040326                          JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5301  017362  004737  037516                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5302  017366  000404                                  BR      140$    ;GO TO 140$ IF NO ERROR
5303  017370  000240                                  NOP                     ;RETURN HERE IF ERROR
5304  017372  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
5305  017374  000137  017460                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5306  017400                                  140$:
5307
5308                                          ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5309  017400  004737  040512                          JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5310  017404  000405                                  BR      150$                    ;GO TO 150$ IF NO ERROR
5311  017406  000240                                  NOP                     ;RETURN HERE IF ERROR
5312  017410  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5313  017412  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5314  017414  000137  017460                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5315  017420                                  150$:
5316  017420  004737  053016                          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5317  017424  000405                                  BR      160$                    ;GO TO 160$ IF NO ERROR
5318  017426  000240                                  NOP                     ;RETURN HERE IF ERROR
5319  017430  104000                                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5320  017432  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5321  017434  000137  017460                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5322  017440                                  160$:
5323  017440  004737  041344                          JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5324  017444  000405                                  BR      170$                    ;GO TO 170$ IF NO ERROR
5325  017446  000240                                  NOP                     ;RETURN HERE IF ERROR
5326  017450  104000                                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5327  017452  004736                                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5328  017454  000137  017460                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5329  017460                                  170$:
5330  017460                                  180$:
5331
5332                                          ;;*****************************************************************
5333                                          ;*TEST 16       FORMAT W/ MID TRANSFER SEEK
5334                                          ;;*****************************************************************
5335  017460                                  TST16:
5336  017460  000004                                  SCOPE                           ;SCOPE CALL
5337  017462  000240                                  NOP                             ;START OF TEST
5338  017464  012706  001100                          MOV     #S^ACK,SP               ;INITIALIZE STACK POINTER
5339  017470  013700  001276                          MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5340  017474  013701  001450                          MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5341  017500  012737  000016  001226                  MOV     #16,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5342  017506                                  10$:
5343
5344                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5345  017506  012737  000000  001434                  MOV     #0,RMDC0                        ;CYLINDER = 0
5346  017514  012737  002037  001406                  MOV     #002037,RMDA0                   ;START AT LAST TRACK & SECTOR
5347  017522  012737  010000  001432                  MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
5348  017530  012737  176774  001402                  MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5349  017536  012737  103566  001404                  MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5350  017544  012737  000062  001400                  MOV     #WH,RMCS1O              ;WRITE HEADER AND DATA
5351  017552  012737  065106  001174                  MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5352  017560  012737  000001  001176                  MOV     #1,$TMP1
```

```
5353  017566  004737  036620            JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5354  017572                    20$:
5355
5356                            ;PREPARE DEVICE FOR DATA TRANSFER
5357  017572  004737  034000            JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5358  017576  154130                    .WORD   154130  ;TASK DESCRIPTOR
5359  017600  000404                    BR      30$                     ;GO TO 30$ IF NO ERROR
5360  017602  000240                    NOP                             ;RETURN HERE IF ERROR
5361  017604  104000                    ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5362  017606  000137  020234            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5363  017612                    30$:
5364
5365                            ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5366  017612  012737  000005  001400      MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5367  017620  012702  001535            MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5368  017624  112722  000006            MOVB    #RMDA,(R2)+
5369  017630  112722  000034            MOVB    #RMDC,(R2)+
5370  017634  112722  000032            MOVB    #RMOF,(R2)+
5371  017640  112722  000000            MOVB    #RMCS1,(R2)+
5372  017644  112722  000200            MOVB    #200,(R2)+
5373  017650  004737  037766            JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5374  017654  000404                    BR      40$     ;GO TO 40$ IF NO ERROR
5375  017656  000240                    NOP                             ;RETURN HERE IF ERROR
5376  017660  104000                    ERROR                           ;ERROR DEFINED BY PUT SUB
5377  017662  000137  020234            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5378  017666                    40$:
5379
5380                            ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TC COMPLETE
5381  017666  004737  037432            JSR     PC,GETSTS               ;SETUP FOR STATUS
5382  017672  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5383  017676                    50$:
5384
5385                            ;GO READ SEEK STATUS
5386  017676  004737  037516            JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5387  017702  000404                    BR      60$     ;GO TO 60$ IF NO ERROR
5388  017704  000240                    NOP                             ;RETURN HERE IF ERROR
5389  017706  104000                    ERROR                           ;ERROR DEFINED BY GET SUB
5390  017710  000137  020234            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5391  017714                    60$:
5392
5393                            ;VERIFY THE RESULTS OF THE SEEK COMMAND
5394  017714  004737  045420            JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
5395  017720  000405                    BR      70$                             ;GO TO 70$ IF NO ERROR
5396  017722  000240                    NOP                             ;RETURN HERE IF ERROR
5397  017724  104000                    ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5398  017726  004736                    JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5399  017730  000137  020234            JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5400  017734                    70$:
5401
5402                            ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5403  017734  012737  000063  001400      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5404  017742  012702  001540            MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5405  017746  112722  000002            MOVB    #RMWC,(R2)+
5406  017752  112722  000004            MOVB    #RMBA,(R2)+
5407  017756  112722  000000            MOVB    #RMCS1,(R2)+
5408  017762  112722  000200            MOVB    #200,(R2)+
```

```
5409  017766  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5410  017772  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5411  017774  000240                          NOP                     ;RETURN HERE IF ERROR
5412  017776  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5413  020000  000137  020234                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5414  020004                          80$:
5415
5416                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5417  020004  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5418  020010  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5419  020014  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5420  020016  000240                          NOP                     ;RETURN HERE IF ERROR
5421  020020  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5422  020022  000137  020234                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5423  020026                          90$:
5424
5425                                  ;VERIFY RESULTS OF WRITE COMMAND
5426  020026  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5427  020032  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5428  020034  000240                          NOP                     ;RETURN HERE IF ERROR
5429  020036  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5430  020040  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5431  020042  000137  020234                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5432  020046                          100$:
5433  020046  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5434  020052  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5435  020054  000240                          NOP                     ;RETURN HERE IF ERROR
5436  020056  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5437  020060  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5438  020062  000137  020234                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5439  020066                          110$:
5440  020066  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5441  020072  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
5442  020074  000240                          NOP                     ;RETURN HERE IF ERROR
5443  020076  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5444  020100  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5445  020102  000137  020234                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5446  020106                          120$:
5447
5448
5449                                  ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5450  020106  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;READ HEADER & DATA COMMAND
5451  020114  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5452  020120  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5453  020122  000240                          NOP                     ;RETURN HERE IF ERROR
5454  020124  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5455  020126  000137  020234                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5456  020132                          130$:
5457
5458                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5459  020132  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5460  020136  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5461  020142  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
5462  020144  000240                          NOP                     ;RETURN HERE IF ERROR
5463  020146  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5464  020150  000137  020234                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5465  020154                                  140$:
5466
5467                                          ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5468  020154  004737  040512                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5469  020160  000405                                BR      150$                    ;GO TO 150$ IF NO ERROR
5470  020162  000240                                NOP                     ;RETURN HERE IF ERROR
5471  020164  104000                                ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5472  020166  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5473  020170  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5474  020174                                  150$:
5475  020174  004737  053016                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5476  020200  000405                                BR      160$                    ;GO TO 160$ IF NO ERROR
5477  020202  000240                                NOP                     ;RETURN HERE IF ERROR
5478  020204  104000                                ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5479  020206  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5480  020210  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5481  020214                                  160$:
5482  020214  004737  041344                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5483  020220  000405                                BR      170$                    ;GO TO 170$ IF NO ERROR
5484  020222  000240                                NOP                     ;RETURN HERE IF ERROR
5485  020224  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5486  020226  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5487  020230  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5488  020234                                  170$:
5489  020234                                  180$:
5490
5491                                          ;;*****************************************************************
5492                                          ;;*TEST 17       FORMAT W/ IMPLIED SEEK
5493                                          ;;*****************************************************************
5494  020234                                  TST17:
5495  020234  000004                                SCOPE                   ;SCOPE CALL
5496  020236  000240                                NOP                     ;START OF TEST
5497  020240  012706  001100                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5498  020244  013700  001276                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5499  020250  013701  001450                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5500  020254  012737  000017  001226                MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5501  020262                                  10$:
5502
5503                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5504  020262  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
5505  020270  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5506  020276  012737  010000  001432                MOV     #FMT16,RMCF0            ;16 BIT FORMAT
5507  020304  012737  176774  001402                MOV     #<-2*C*<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5508  020312  012737  103566  001404                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5509  020320  012737  000062  001400                MOV     #WH,RMCS1O              ;WRITE HEADER AND DATA
5510  020326  012737  065106  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5511  020334  012737  000001  001176                MOV     #1,$TMP1
5512  020342  004737  036620                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5513  020346                                  20$:
5514
5515                                          ;PREPARE DEVICE FOR DATA TRANSFER
5516  020346  004737  034000                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5517  020352  154130                        .WORD   154130  ;TASK DESCRIPTOR
5518  020354  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
5519  020356  000240                        NOP                     ;RETURN HERE IF ERROR
5520  020360  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
5521  020362  000137  021032                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5522  020366                                30$:
5523
5524                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE OFF CYLINDER
5525  020366  013737  001434  021034                MOV     RMDC0,190$              ;SAVE CYLINDER ADDRESS
5526  020374  012737  001466  001434                MOV     #822.,RMDC0             ;SEEK TO LAST CYLINDER
5527  020402  012737  000005  001400                MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5528  020410  012702  001535                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5529  020414  112722  000006                        MOVB    #RMDA,(R2)+
5530  020420  112722  000034                        MOVB    #RMDC,(R2)+
5531  020424  112722  000032                        MOVB    #RMOF,(R2)+
5532  020430  112722  000000                        MOVB    #RMCS1,(R2)+
5533  020434  112722  000200                        MOVB    #200,(R2)+
5534  020440  004737  037766                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5535  020444  000404                                BR      40$     ;GO TO 40$ IF NO ERROR
5536  020446  000240                                NOP                             ;RETURN HERE IF ERROR
5537  020450  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
5538  020452  000137  021032                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5539  020456                                40$:
5540
5541                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5542  020456  004737  037432                        JSR     PC,GETSTS               ;SETUP FOR STATUS
5543  020462  004737  040326                        JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5544  020466                                50$:
5545
5546
5547  020466  004737  037516                ;GO READ SEEK STATUS
                                                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5548  020472  000404                                BR      60$     ;GO TO 60$ IF NO ERROR
5549  020474  000240                                NOP                             ;RETURN HERE IF ERROR
5550  020476  104000                                ERROR                           ;ERROR DEFINED BY GET SUB
5551  020500  000137  021032                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5552  020504                                60$:
5553
5554                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
5555  020504  004737  045420                        JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
5556  020510  000405                                BR      70$                             ;GO TO 70$ IF NO ERROR
5557  020512  000240                                NOP                             ;RETURN HERE IF ERROR
5558  020514  104000                                ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5559  020516  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5560  020520  000137  021032                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5561  020524                                70$:
5562
5563                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5564  020524  013737  021034  001434                MOV     190$,RMDC0              ;RESTORE DISK ADDRESS
5565  020532  012737  000063  001400                MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5566  020540  012702  001540                        MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5567  020544  112722  000002                        MOVB    #RMWC,(R2)+
5568  020550  112722  000004                        MOVB    #RMBA,(R2)+
5569  020554  112722  000000                        MOVB    #RMCS1,(R2)+
5570  020560  112722  000200                        MOVB    #200,(R2)+
5571  020564  004737  037766                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5572  020570  000404                                BR      80$     ;GO TO 80$ IF NO ERROR
5573  020572  000240                                NOP                             ;RETURN HERE IF ERROR
5574  020574  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
5575  020576  000137  021032                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5576  020602                                80$:
```

```
                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
020602  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
020606  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
020612  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
020614  000240                          NOP                     ;RETURN HERE IF ERROR
020616  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
020620  000137  021032                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
020624                          90$:

                                ;VERIFY RESULTS OF WRITE COMMAND
020624  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
020630  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
020632  000240                          NOP                     ;RETURN HERE IF ERROR
020634  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
020636  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020640  000137  021032                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
020644                          100$:
020644  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
020650  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
020652  000240                          NOP                     ;RETURN HERE IF ERROR
020654  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
020656  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020660  000137  021032                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
020664                          110$:
020664  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
020670  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
020672  000240                          NOP                     ;RETURN HERE IF ERROR
020674  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
020676  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020700  000137  021032                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
020704                          120$:

                                ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
020704  012737  000053  001400          MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
020712  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
020716  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
020720  000240                          NOP                     ;RETURN HERE IF ERROR
020722  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
020724  000137  021032                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
020730                          130$:

                                ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
020730  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
020734  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
020740  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
020742  000240                          NOP                     ;RETURN HERE IF ERROR
020744  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
020746  000137  021032                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
020752                          140$:

                                ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
020752  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
020756  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
020760  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
5633  020762  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5634  020764  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5635  020766  000137  021032                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5636  020772                          150$:
5637  020772  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5638  020776  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
5639  021000  000240                          NOP                             ;RETURN HERE IF ERROR
5640  021002  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5641  021004  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5642  021006  000137  021032                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5643  021012                          160$:
5644  021012  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5645  021016  000405                          BR      170$                            ;GO TO 170$ IF NO ERROR
5646  021020  000240                          NOP                             ;RETURN HERE IF ERROR
5647  021022  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5648  021024  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5649  021026  000137  021032                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5650  021032                          170$:
5651  021032                          180$:
5652  021032  000401                          BR      200$
5653  021034  000000                  190$:   .WORD                                   ;TEMPORARY STORAGE
5654  021036                          200$:
5655
5656                                  ;;*****************************************************************
5657                                  ;*TEST 20        FORMAT EACH SECTOR ADDRESS
5658                                  ;;*****************************************************************
5659  021036                          TST20:
5660  021036  000004                          SCOPE                           ;SCOPE CALL
5661  021040  000240                          NOP                             ;START OF TEST
5662  021042  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
5663  021046  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5664  021052  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5665  021056  012737  000020  001226          MOV     #20,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5666  021064                          10$:
5667
5668                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5669  021064  012737  000000  001434          MOV     #0,RMDC0                        ;CYLINDER = 0
5670  021072  012737  000000  001406          MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
5671  021100  012737  010000  001432  15$:     MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5672  021106  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
5673  021114  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5674  021122  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5675  021130  012737  001406  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
5676  021136  012737  000001  001176          MOV     #1,$TMP1
5677  021144  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5678  021150                          20$:
5679
5680                                  ;PREPARE DEVICE FOR DATA TRANSFER
5681  021150  004737  034000                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5682  021154  154130                          .WORD   154130  ;TASK DESCRIPTOR
5683  021156  000404                          BR      30$                             ;GO TO 30$ IF NO ERROR
5684  021160  000240                          NOP                             ;RETURN HERE IF ERROR
5685  021162  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5686  021164  000137  021562                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
5687  021170                          30$:
5688
```

```
5689                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5690  021170  012737  000063  001400            MOV     #WH!GO,RMCS1O           ;WRITE HEADER AND DATA
5691  021176  012702  001536                    MOV     #WRITINX,R2             ;WRITE REGISTER INDEX TABLE
5692  021202  112723  000006                    MOVB    #6,(R2)+
5693  021206  112722  000034                    MOVB    #34,(R2)+
5694  021212  112722  000032                    MOVB    #RMOF,(R2)+
5695  021216  112723  000002                    MOVB    #2,(R2)+
5696  021222  112722  000004                    MOVB    #4,(R2)+
5697  021226  112722  000000                    MOVB    #RMCS1,(R2)+
5698  021232  112722  000200                    MOVB    #200,(R2)+
5699  021236  004737  037766    ;               JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5700  021242  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5701  021244  000240                            NOP                             ;RETURN HERE IF ERROR
5702  021246  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
5703  021250  000137  021562                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5704  021254                          80$:
5705
5706                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
5707  021254  004737  037432                    JSR     PC,GETSTS
5708
5709                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5710  021260  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5711  021264  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5712  021270  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5713  021272  000240                            NOP                             ;RETURN HERE IF ERROR
5714  021274  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5715  021276  000137  021562                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5716  021302                          90$:
5717
5718                                    ;VERIFY RESULTS OF WRITE COMMAND
5719  021302  004737  040512                    JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5720  021306  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
5721  021310  000240                            NOP                             ;RETURN HERE IF ERROR
5722  021312  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5723  021314  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5724  021316  000137  021562                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5725  021322                          100$:
5726  021322  004737  053016                    JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5727  021326  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
5728  021330  000240                            NOP                             ;RETURN HERE IF ERROR
5729  021332  104000                            ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5730  021334  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5731  021336  000137  021562                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5732  021342                          110$:
5733  021342  004737  041344                    JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5734  021346  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
5735  021350  000240                            NOP                             ;RETURN HERE IF ERROR
5736  021352  104000                            ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5737  021354  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5738  021356  000137  021562                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5739  021362                          120$:
5740
5741
5742                                    ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5743  021362  012737  000073  001400            MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
5744  021370  012737  104572  001404            MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
```

```
5745  021376  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5746  021402  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5747  021404  000240                          NOP                     ;RETURN HERE IF ERROR
5748  021406  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5749  021410  000137  021562                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5750  021414                          130$:
5751
5752                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
5753  021414  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5754  021420  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5755  021424  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
5756  021426  000240                          NOP                     ;RETURN HERE IF ERROR
5757  021430  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5758  021432  000137  021562                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5759  021436                          140$:
5760
5761                                  ;VERIFY THE RESULTS OF READ OPERATION
5762  021436  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5763  021442  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5764  021444  000240                          NOP                     ;RETURN HERE IF ERROR
5765  021446  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5766  021450  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5767  021452  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5768  021456                          150$:
5769  021456  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5770  021462  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5771  021464  000240                          NOP                     ;RETURN HERE IF ERROR
5772  021466  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5773  021470  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5774  021472  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5775  021476                          160$:
5776  021476  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5777  021502  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
5778  021504  000240                          NOP                     ;RETURN HERE IF ERROR
5779  021506  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5780  021510  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5781  021512  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5782  021516                          170$:
5783
5784                                  ;VERIFY DATA
5785  021516  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5786  021522  103566                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
5787  021524  104572                          .WORD   BUFTWO          ;STARTING ADDRESS OF READ BUFFER
5788  021526  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
5789  021530  000240                          NOP                     ;RETURN HERE IF ERROR
5790  021532  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5791  021534  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5792  021540                          180$:
5793
5794                                  ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
5795  021540  062737  000001  001406          ADD     #1,RMDA0                ;ADVANCE SECTOR COUNT
5796  021546  122737  000037  001406          CMPB    #31.,RMDA0              ;DONE ALL SECTORS??
5797  021554  103402                          BLO     190$                    ;YES!!
5798  021556  000137  021100                  JMP     15$                     ;GO DO NEXT SECTOR
5799  021562                          190$:
5800
```

```
                                ;;*****************************************************************
                                ;*TEST 21       FORMAT EACH TRACK ADDRESS
                                ;*****************************************************************
021562                          TST21:
021562  000004                          SCOPE                   ;SCOPE CALL
021564  000240                          NOP                     ;START OF TEST
021566  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
021572  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
021576  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
021602  012737  000021  001226          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021610                          10$:
                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
021610  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
021616  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
021624  012737  010000  001432  15$:     MOV     #FMT16,RMOF0            ;16 BIT FORMAT
021632  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
021640  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
021646  012737  000063  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
021654  012737  001436  001174          MOV     #RMDA0,$TMP0            ;USE TRACK FOR DATA PATTERN
021662  012737  000001  001176          MOV     #1,$TMP1
021670  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
021674                          20$:
                                ;PREPARE DEVICE FOR DATA TRANSFER
021674  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
021700  154130                  .WORD   154130  ;TASK DESCRIPTOR
021702  000404                  BR      30$                     ;GO TO 30$ IF NO ERROR
021704  000240                  NOP                             ;RETURN HERE IF ERROR
021706  104000                  ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
021710  000137  022306          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
021714                          30$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
021714  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
021722  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
021726  112722  000006                  MOVB    #RMDA,(R2)+
021732  112722  000034                  MOVB    #RMDC,(R2)+
021736  112722  000032                  MOVB    #RMOF,(R2)+
021742  112722  000002                  MOVB    #RMWC,(R2)+
021746  112722  000004                  MOVB    #RMBA,(R2)+
021752  112722  000000                  MOVB    #RMCS1,(R2)+
021756  112722  000200                  MOVB    #200,(R2)+
021762  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
021766  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
021770  000240                  NOP                             ;RETURN HERE IF ERROR
021772  104000                  ERROR                           ;ERROR DEFINED BY PUT SUB
021774  000137  022306          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
022000                          80$:

                                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
022000  004737  037432                  JSR     PC,GETSTS

                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
022004  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
022010  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
```

```
5857  022014  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5858  022016  000240                        NOP                     ;RETURN HERE IF ERROR
5859  022020  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5860  022022  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5861  022026                        90$:
5862
5863                                  ;VERIFY RESULTS OF WRITE COMMAND
5864  022026  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5865  022032  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5866  022034  000240                        NOP                     ;RETURN HERE IF ERROR
5867  022036  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5868  022040  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5869  022042  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5870  022046                        100$:
5871  022046  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5872  022052  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5873  022054  000240                        NOP                     ;RETURN HERE IF ERROR
5874  022056  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5875  022060  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5876  022062  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5877  022066                        110$:
5878  022066  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5879  022072  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
5880  022074  000240                        NOP                     ;RETURN HERE IF ERROR
5881  022076  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBPOUTINE
5882  022100  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5883  022102  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5884  022106                        120$:
5885
5886
5887                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5888  022106  012737  000073  001400        MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
5889  022114  012737  104572  001404        MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
5890  022122  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5891  022126  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
5892  022130  000240                        NOP                     ;RETURN HERE IF ERROR
5893  022132  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
5894  022134  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5895  022140                        130$:
5896
5897                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
5898  022140  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5899  022144  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5900  022150  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
5901  022152  000240                        NOP                     ;RETURN HERE IF ERROR
5902  022154  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5903  022156  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5904  022162                        140$:
5905
5906                                  ;VERIFY THE RESULTS OF READ OPERATION
5907  022162  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5908  022166  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5909  022170  000240                        NOP                     ;RETURN HERE IF ERROR
5910  022172  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5911  022174  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5912  022176  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
```

```
5913  022202                          150$:
5914  022202  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5915  022206  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5916  022210  000240                          NOP                     ;RETURN HERE IF ERROR
5917  022212  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5918  022214  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5919  022216  000137  022306                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5920  022222                          160$:
5921  022222  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5922  022226  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
5923  022230  000240                          NOP                     ;RETURN HERE IF ERROR
5924  022232  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5925  022234  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5926  022236  000137  022306                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5927  022242                          170$:
5928
5929                                  ;VERIFY DATA
5930  022242  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5931  022246  103566                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
5932  022250  104572                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
5933  022252  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
5934  022254  000240                          NOP                     ;RETURN HERE IF ERROR
5935  022256  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5936  022260  000137  022306                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5937  022264                          180$:
5938
5939                                  ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
5940  022264  062737  000400  001406          ADD     #400,RMDAO              ;ADVANCE TRACK COUNT
5941  022272  022737  002000  001406          CMP     #2000,RMDAO             ;DONE ALL TRACKS??
5942  022300  103402                          BLO     190$                    ;YES!!
5943  022302  000137  021624                  JMP     15$                     ;GO DO NEXT SECTOR
5944  022306                          190$:
5945
5946                                  ;;*****************************************************************
5947                                  ;*TEST 22        FORMAT PRIME CYLINDERS
5948                                  ;;*****************************************************************
5949  022306                          TST22:
5950  022306  000004                          SCOPE                   ;SCOPE CALL
5951  022310  000240                          NOP                     ;START OF TEST
5952  022312  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5953  022316  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5954  022322  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5955  022326  012737  000022  001226          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5956  022334                          10$:
5957
5958                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5959  022334  012737  000001  001434          MOV     #1,RMDC0                ;CYLINDER = 0
5960  022342  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5961  022350  012737  010000  001432  15$:     MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5962  022356  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
5963  022364  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5964  022372  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5965  022400  012737  001434  001174          MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
5966  022406  012737  000001  001176          MOV     #1,$TMP1
5967  022414  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5968  022420                          20$:
```

```
5969
5970                                    ;PREPARE DEVICE FOR DATA TRANSFER
5971  022420  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5972  022424  154130                            .WORD   154130  ;TASK DESCRIPTOR
5973  022426  000404                            BR      20$             ;GO TO 30$ IF NO ERROR
5974  022430  000240                            NOP                     ;RETURN HERE IF ERROR
5975  022432  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5976  022434  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5977  022440                            30$:
5978
5979                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5980  022440  012737  000063  001400            MOV     #WHDGO!RMCS10           ;WRITE HEADER AND DATA
5981  022446  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5982  022452  112722  000006                    MOVB    #RMDA,(R2)+
5983  022456  112722  000034                    MOVB    #RMDC,(R2)+
5984  022462  112722  000032                    MOVB    #RMOF,(R2)+
5985  022466  112722  000002                    MOVB    #RMWC,(R2)+
5986  022472  112722  000004                    MOVB    #RMBA,(R2)+
5987  022476  112722  000000                    MOVB    #RMCS1,(R2)+
5988  022502  112722  000200                    MOVB    #200,(R2)+
5989  022506  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5990  022512  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5991  022514  000240                            NOP                     ;RETURN HERE IF ERROR
5992  022516  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5993  022520  000137  023030                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5994  022524                            80$:
5995
5996                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
5997  022524  004737  037432                    JSR     PC,GETSTS
5998
5999                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6000  022530  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6001  022534  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6002  022540  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
6003  022542  000240                            NOP                     ;RETURN HERE IF ERROR
6004  022544  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
6005  022546  000137  023030                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6006  022552                            90$:
6007
6008                                    ;VERIFY RESULTS OF WRITE COMMAND
6009  022552  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6010  022556  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
6011  022560  000240                            NOP                     ;RETURN HERE IF ERROR
6012  022562  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6013  022564  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6014  022566  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6015  022572                            100$:
6016  022572  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6017  022576  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
6018  022600  000240                            NOP                     ;RETURN HERE IF ERROR
6019  022602  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6020  022604  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6021  022606  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6022  022612                            110$:
6023  022612  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6024  022616  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
```

```
6025  022620  000240                          NOP                     ;RETURN HERE IF ERROR
6026  022622  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6027  022624  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6028  022626  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6029  022632                          120$:
6030
6031
6032                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6033  022632  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
6034  022640  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
6035  022646  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6036  022652  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6037  022654  000240                          NOP                     ;RETURN HERE IF ERROR
6038  022656  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6039  022660  000137  023030                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6040  022664                          130$:
6041
6042                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6043  022664  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6044  022670  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6045  022674  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6046  022676  000240                          NOP                     ;RETURN HERE IF ERROR
6047  022700  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6048  022702  000137  023030                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6049  022706                          140$:
6050
6051                                  ;VERIFY THE RESULTS OF READ OPERATION
6052  022706  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6053  022712  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6054  022714  000240                          NOP                     ;RETURN HERE IF ERROR
6055  022716  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6056  022720  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6057  022722  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6058  022726                          150$:
6059  022726  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6060  022732  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6061  022734  000240                          NOP                     ;RETURN HERE IF ERROR
6062  022736  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6063  022740  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6064  022742  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6065  022746                          160$:
6066  022746  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6067  022752  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6068  022754  000240                          NOP                     ;RETURN HERE IF ERROR
6069  022756  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6070  022760  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6071  022762  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6072  022766                          170$:
6073
6074                                  ;VERIFY DATA
6075  022766  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
6076  022772  103566                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
6077  022774  104572                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
6078  022776  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
6079  023000  000240                          NOP                     ;RETURN HERE IF ERROR
6080  023002  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
```

```
6081  023004  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6082  023010                          180$:
6083
6084                                  ;INCREMENT ADDRESS AND FORMAT NEXT PRIME CYLINDER
6085  023010  006337  001434                  ASL     RMDCO                   ;ADVANCE CYLINDER COUNT
6086  023014  022737  001466  001434          CMP     #822.,RMDCO             ;DONE ALL CYLINDERS??
6087  023022  103402                          BLO     190$                    ;YES!!
6088  023024  000137  022350                  JMP     15$                     ;GO DO NEXT CYLINDER
6089  023030                          190$:
6090
6091                                  ;;*****************************************************************
6092                                  ;*TEST 23        FORMAT LAST SECTOR
6093                                  ;;*****************************************************************
6094  023030                          TST23:
6095  023030  000004                          SCOPE                           ;SCOPE CALL
6096  023032  000240                          NOP                             ;START OF TEST
6097  023034  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6098  023040  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6099  023044  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6100  023050  012737  000023  001226          MOV     #23,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6101  023056                          10$:
6102
6103                                  ;PREPARE DEVICE FOR DATA TRANSFER
6104  023056  004737  034000                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6105  023062  154130                          .WORD   154130  ;TASK DESCRIPTOR
6106  023064  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6107  023066  000240                          NOP                             ;RETURN HERE IF ERROR
6108  023070  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6109  023072  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6110  023076                          30$:
6111
6112                                  ;SETUP PARAMETERS FOR READING LAST SECTOR
6113  023076  012737  001466  001434          MOV     #822.,RMDCO             ;LAST CYLINDER
6114  023104  012737  002037  001406          MOV     #2037,RMDAO             ;TRACK = 4 SECTOR = 31.
6115  023112  012737  010000  001432          MOV     #FMT16,RMOFO            ;16 BIT FORMAT
6116  023120  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWCO   ;2 + 256 WORDS
6117  023126  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;DATA BUFFER ADDRESS
6118  023134  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;WRITE HEADER AND DATA
6119
6120  023142  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6121  023146  112722  000006                  MOVB    #RMDA,(R2)+
6122  023152  112722  000034                  MOVB    #RMOC,(R2)+
6123  023156  112722  000032                  MOVB    #RMOF,(R2)+
6124  023162  112722  000004                  MOVB    #RMBA,(R2)+
6125  023166  112722  000002                  MOVB    #RMWC,(R2)+
6126  023172  112722  000000                  MOVB    #RMCS1,(R2)+
6127  023176  112722  000200                  MOVB    #200,(R2)+
6128
6129
6130  023202                          120$:
6131
6132                                  ;READ HEADER AND DATA OF LAST SECTOR
6133  023202  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6134  023206  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6135  023210  000240                          NOP                             ;RETURN HERE IF ERROR
6136  023212  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
```

```
6137  023214  000137  023326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6138  023220                          130$:
6139
6140                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6141  023220  004737  037432                  JSR     PC,GETSTS
6142                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6143  023224  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6144  023230  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6145  023234  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6146  023236  000240                          NOP                             ;RETURN HERE IF ERROR
6147  023240  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
6148  023242  000137  023326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6149  023246                          140$:
6150
6151                                  ;VERIFY THE RESULTS OF READ OPERATION
6152  023246  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6153  023252  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6154  023254  000240                          NOP                             ;RETURN HERE IF ERROR
6155  023256  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6156  023260  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6157  023262  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6158  023266                          150$:
6159  023266  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6160  023272  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6161  023274  000240                          NOP                             ;RETURN HERE IF ERROR
6162  023276  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6163  023300  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6164  023302  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6165  023306                          160$:
6166  023306  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6167  023312  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6168  023314  000240                          NOP                             ;RETURN HERE IF ERROR
6169  023316  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6170  023320  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6171  023322  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6172  023326                          170$:
6173
6174  023326                          190$:
6175
6176                                  ;;*****************************************************************
6177                                  ;*TEST 24        FORMAT W/ AOE ERROR
6178                                  ;;*****************************************************************
6179  023326                          TST24:
6180  023326  000004                          SCOPE                           ;SCOPE CALL
6181  023330  000240                          NOP                             ;START OF TEST
6182  023332  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6183  023336  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6184  023342  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6185  023346  012737  000024  001226          MOV     #24,$TEST#              ;;SET TEST NUMBER IN APT MAIL BOX
6186
6187                                  ;PREPARE DEVICE FOR DATA TRANSFER
6188  023354  004737  034000                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6189  023360  154130                          .WORD   154130  ;TASK DESCRIPTOR
6190  023362  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6191  023364  000240                          NOP                             ;RETURN HERE IF ERROR
6192  023366  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
023370  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
023374                          30$:

                                ;SETUP PARAMETERS FOR READING 2 SECTORS STARTING WITH LAST SECTOR
023374  012737  001466  001434          MOV     #822.,RMDC0             ;START AT LAST CYLINDER
023402  012737  002037  001406          MOV     #2037,RMDA0             ;TRACK = 4 SECTOR = 31.
023410  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
023416  012737  176774  001402          MOV     #<^C<<2+256.>*2>+1>,RMWC0       ;FORMAT 2 SECTORS
023424  012737  104572  001404          MOV     #BUFTW0,RMBA0           ;DATA BUFFER ADDRESS
023432  012737  000073  001400          MOV     #RH!GO,RMCS10           ;WRITE HEADER AND DATA
023440  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
023444  112722  000006                  MOVB    #RMOA,(R2)+
023450  112722  000034                  MOVB    #RMDC,(R2)+
023454  112722  000032                  MOVB    #RMOF,(R2)+
023460  112722  000002                  MOVB    #RMWC,(R2)+
023464  112722  000004                  MOVB    #RMBA,(R2)+
023470  112722  000000                  MOVB    #RMCS1,(R2)+
023474  112722  000200                  MOVB    #200,(R2)+
023500                          80$:

023500                          125$:
023500  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
023504  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
023506  000240                          NOP                             ;RETURN HERE IF ERROR
023510  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
023512  000137  023624                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
023516                          130$:

                                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
023516  004737  037432                  JSR     PC,GETSTS

                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
023522  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
023526  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
023532  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
023534  000240                          NOP                             ;RETURN HERE IF ERROR
023536  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
023540  000137  023624                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
023544                          140$:

                                ;VERIFY THE RESULTS OF READ OPERATION
023544  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
023550  000405                          BR      150$                            ;GO TO 150$ IF NO ERROR
023552  000240                          NOP                             ;RETURN HERE IF ERROR
023554  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
023556  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
023560  000137  023624                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
023564                          150$:
023564  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
023570  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
023572  000240                          NOP                             ;RETURN HERE IF ERROR
023574  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
023576  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
023600  000137  023624                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
023604                          160$:
023604  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
```

```
6249  023610  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6250  023612  000240                          NOP                     ;RETURN HERE IF ERROR
6251  023614  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6252  023616  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6253  023620  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6254  023624                          170$:
6255
6256  023624                          190$:
6257
6258                                  ;;*****************************************************************
6259                                  ;*TEST 25        FORMAT INVALID SECTOR ADDRESS
6260                                  ;;*****************************************************************
6261  023624                          TST25:
6262  023624  000004                          SCOPE                   ;SCOPE CALL
6263  023626  000240                          NOP                     ;START OF TEST
6264  023630  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6265  023634  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6266  023640  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6267  023644  012737  000025  001226          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6268  023652                          10$:
6269
6270                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6271  023652  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
6272  023660  012737  000036  001406          MOV     #30.,RMDA0              ;TRACK = 0, SECTOR = 30
6273  023666  012737  000000  001432  15$:     MOV     #0,RMOF0                ;18 BIT FORMAT
6274  023674  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6275  023702  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6276  023710  012737  000062  001400          MOV     #WH.RMCS10              ;WRITE HEADER AND DATA
6277  023716  012737  001406  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6278  023724  012737  000001  001176          MOV     #1,$TMP1
6279  023732  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6280  023736                          20$:
6281
6282                                  ;PREPARE DEVICE FOR DATA TRANSFER
6283  023736  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6284  023742  154130                          .WORD   154130  ;TASK DESCRIPTOR
6285  023744  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6286  023746  000240                          NOP                     ;RETURN HERE IF ERROR
6287  023750  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6288  023752  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6289  023756                          30$:
6290
6291                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6292  023756  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6293  023764  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6294  023770  112722  000006                  MOVB    #RMDA,(R2)+
6295  023774  112722  000034                  MOVB    #RMDC,(R2)+
6296  024000  112722  000032                  MOVB    #RMOF,(R2)+
6297  024004  112722  000002                  MOVB    #RMWC,(R2)+
6298  024010  112722  000004                  MOVB    #RMBA,(R2)+
6299  024014  112722  000000                  MOVB    #RMCS1,(R2)+
6300  024020  112722  000200                  MOVB    #200,(R2)+
6301  024024  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6302  024030  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6303  024032  000240                          NOP                     ;RETURN HERE IF ERROR
6304  024034  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
```

```
                        T25     FORMAT INVALID SECTOR ADDRESS

6305  024036  000137  024352            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6306  024042                    80$:
6307
6308                              ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6309  024042  004737  037432            JSR     PC,GETSTS
6310
6311                              ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6312  024046  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6313  024052  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6314  024056  000404                    BR      90$     ;GO TO 90$ IF NO ERROR
6315  024060  000240                    NOP                     ;RETURN HERE IF ERROR
6316  024062  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
6317  024064  000137  024352            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6318  024070                    90$:
6319
6320                              ;VERIFY RESULTS OF WRITE COMMAND
6321  024070  004737  040512            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6322  024074  000405                    BR      100$                    ;GO TO 100$ IF NO ERROR
6323  024076  000240                    NOP                     ;RETURN HERE IF ERROR
6324  024100  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6325  024102  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6326  024104  000137  024352            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6327  024110                    100$:
6328  024110  004737  053016            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6329  024114  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
6330  024116  000240                    NOP                     ;RETURN HERE IF ERROR
6331  024120  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6332  024122  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6333  024124  000137  024352            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6334  024130                    110$:
6335  024130  004737  041344            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6336  024134  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
6337  024136  000240                    NOP                     ;RETURN HERE IF ERROR
6338  024140  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6339  024142  004736                    JSP     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6340  024144  000137  024352            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6341  024150                    120$:
6342
6343                              ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6344  024150  004737  034000            JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6345  024154  154130                    .WORD   154130  ;TASK DESCRIPTOR
6346  024156  000404                    BR      125$                    ;GO TO 125$ IF NO ERROR
6347  024160  000240                    NOP                     ;RETURN HERE IF ERROR
6348  024162  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6349  024164  000137  024352            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6350  024170                    125$:
6351
6352                              ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6353  024170  012737  000073  001400      MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
6354  024176  012737  104572  001404      MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
6355  024204  004737  037766            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6356  024210  000404                    BR      130$    ;GO TO 130$ IF NO ERROR
6357  024212  000240                    NOP                     ;RETURN HERE IF ERROR
6358  024214  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
6359  024216  000137  024352            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6360  024222                    130$:
```

```
6361
6362
6363  024222  004737  037432          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
                                              JSR     PC,GETSTS
6364
6365                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6366  024226  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6367  024232  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6368  024236  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
6369  024240  000240                        NOP                             ;RETURN HERE IF ERROR
6370  024242  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
6371  024244  000137  024352                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6372  024250                        140$:
6373
6374                                  ;VERIFY THE RESULTS OF READ OPERATION
6375  024250  004737  040512                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6376  024254  000405                        BR      150$                            ;GO TO 150$ IF NO ERROR
6377  024256  000240                        NOP                             ;RETURN HERE IF ERROR
6378  024260  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6379  024262  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6380  024264  000137  024352                JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
6381  024270                        150$:
6382  024270  004737  053016                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6383  024274  000405                        BR      160$                            ;GO TO 160$ IF NO ERROR
6384  024276  000240                        NOP                             ;RETURN HERE IF ERROR
6385  024300  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6386  024302  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6387  024304  000137  024352                JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
6388  024310                        160$:
6389  024310  004737  041344                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6390  024314  000405                        BR      170$                            ;GO TO 170$ IF NO ERROR
6391  024316  000240                        NOP                             ;RETURN HERE IF ERROR
6392  024320  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6393  024322  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6394  024324  000137  024352                JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
6395  024330                        170$:
6396  024330                        180$:
6397
6398                                  ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
6399  024330  062737  000001  001406        ADD     #1,RP.2                 ;ADVANCE SECTOR COUNT
6400  024336  022737  000037  001406        CMP     #31.,.JAO               ;DONE ALL SECTORS??
6401  024344  103402                        BLO     190$                    ;YES!!
6402  024346  000137  023666                JMP     15$                     ;GO DO NEXT SECTOR
6403  024352                        190$:
6404
6405                                  ;;*****************************************************************
6406                                  ;*TEST 26        FORMAT INVALID TRACK ADDRESS
6407                                  ;;*****************************************************************
6408  024352                        TST26:
6409  024352  000004                        SCOPE                           ;SCOPE CALL
6410  024354  000240                        NOP                             ;START OF TEST
6411  024356  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6412  024362  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6413  024366  013701  001450                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6414  024372  012737  000026  001226        MOV     #26,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6415  024400                        10$:
6416
```

```
6417                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6418  024400  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
6419  024406  012737  002400  001406            MOV     #2400,RMDA0             ;TRACK = 5,SECTOR = 0
6420  024414  012737  010000  001432  15$:      MOV     #FMT16,RMFO             ;16 BIT FORMAT
6421  024422  012737  177376  001402            MOV     #<^C<2+2^ .>+1>,RMWC0   ;2 + 256 WORDS
6422  024430  012737  103566  001404            MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6423  024436  012737  000062  001400            MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6424  024444  012737  001406  001174            MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6425  024452  012737  000001  001176            MOV     #1,$TMP1
6426  024460  004737  036620                    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6427  024464                            20$:
6428
6429                                    ;PREPARE DEVICE FOR DATA TRANSFER
6430  024464  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6431  024470  154130                            .WORD   154130  ;TASK DESCRIPTOR
6432  024472  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
6433  024474  000240                            NOP                     ;RETURN HERE IF ERROR
6434  024476  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6435  024500  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6436  024504                            30$:
6437
6438                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6439  024504  012737  000063  001400            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6440  024512  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6441  024516  112722  000006                    MOVB    #RMDA,(R2)+
6442  024522  112722  000034                    MOVB    #RMDC,(R2)+
6443  024526  112722  000032                    MOVB    #RMOF,(R2)+
6444  024532  112722  000103                    MOVB    #RMWC,(R2)+
6445  024536  112722  000004                    MOVB    #RMBA,(R2)+
6446  024542  112722  000000                    MOVB    #RMCS1,(R2)+
6447  024546  112722  000200                    MOVB    #200,(R2)+
6448  024552  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6449  024556  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
6450  024560  000240                            NOP                     ;RETURN HERE IF ERROR
6451  024562  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
6452  024564  000137  025100                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6453  024570                            80$:
6454
6455                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6456  024570  004737  037432                    JSR     PC,GETSTS
6457
6458                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6459  024574  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6460  024600  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6461  024604  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
6462  024606  000240                            NOP                     ;RETURN HERE IF ERROR
6463  024610  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
6464  024612  000137  025100                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6465  024616                            90$:
6466
6467                                    ;VERIFY RESULTS OF WRITE COMMAND
6468  024616  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6469  024622  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
6470  024624  000240                            NOP                     ;RETURN HERE IF ERROR
6471  024626  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6472  024630  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
6473  024632  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6474  024636                            100$:
6475  024636  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6476  024642  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
6477  024644  000240                            NOP                     ;RETURN HERE IF ERROR
6478  024646  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6479  024650  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6480  024654  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6481  024660                            110$:
6482  024660  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6483  024664  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
6484  024666  000240                            NOP                     ;RETURN HERE IF ERROR
6485  024670  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6486  024672  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6487  024676  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6488  024702                            120$:
6489
6490                                    ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6491  024702  004737  034000                    JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6492  024706  154130                            .WORD   154130  ;TASK DESCRIPTOR
6493  024710  000404                            BR      125$                    ;GO TO 125$ IF NO ERROR
6494  024712  000240                            NOP                     ;RETURN HERE IF ERROR
6495  024714  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6496  024716  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6497  024722                            125$:
6498
6499                                    ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6500  024722  012737  000073  001400            MOV     #RH!GO,RMCS10                   ;READ HEADER & DATA COMMAND
6501  024730  012737  104572  001404            MOV     #BUFTWO,RMBAO                   ;CHANGE BUS ADDRESS
6502  024736  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6503  024742  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
6504  024744  000240                            NOP                     ;RETURN HERE IF ERROR
6505  024746  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
6506  024750  000137  025100                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6507  024754                            130$:
6508
6509                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6510  024754  004737  037432                    JSR     PC,GETSTS
6511
6512                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
6513  024760  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6514  024764  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6515  024770  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
6516  024772  000240                            NOP                     ;RETURN HERE IF ERROR
6517  024774  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
6518  024776  000137  025100                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6519  025002                            140$:
6520
6521                                    ;VERIFY THE RESULTS OF READ OPERATION
6522  025002  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6523  025006  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
6524  025010  000240                            NOP                     ;RETURN HERE IF ERROR
6525  025012  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6526  025014  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6527  025020  000137  025100                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6528  025024                            150$:
```

```
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      160$                    ;GO TO 160$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
                                160$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      170$                    ;GO TO 170$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
                                170$:
                                180$:

                                ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
                                        ADD     #400,RMDA0              ;ADVANCE TRACK COUNT
                                        CMP     #3400,RMDA0             ;DONE ALL TRACKS??
                                        BLO     190$                    ;YES!!
                                        JMP     15$                     ;GO DO NEXT TRACK
                                190$:

                                ;;*****************************************************************
                                ;*TEST 27       FORMAT INVALID CYLINDER ADDRESS
                                ;;*****************************************************************
                                TST27:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTUNI,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
                                        MOV     #823.,RMDC0             ;START AT LAST CYLINDER + 1
                                        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
                                15$:    MOV     #FMT16,RMDF0            ;16 BIT FORMAT
                                        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
                                        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
                                        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
                                        MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
                                        MOV     #1,$TMP1
                                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
                                20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   154130  ;TASK DESCRIPTOR
                                        BR      30$                     ;GO TO 30$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
                                30$:
```

```
6585                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6586  025232  012737  000063  001400            MOV     #WHIGO,RMCS1O           ;WRITE HEADER AND DATA
6587  025240  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6588  025244  112722  000006                    MOVB    #RMDA,(R2)+
6589  025250  112722  000034                    MOVB    #RMDC,(R2)+
6590  025254  112722  000032                    MOVB    #RMOF,(R2)+
6591  025260  112722  000002                    MOVB    #RMWC,(R2)+
6592  025264  112722  000004                    MOVB    #RMBA,(R2)+
6593  025270  112722  000000                    MOVB    #RMCS1,(R2)+
6594  025274  112722  000200                    MOVB    #200,(R2)+
6595  025300  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6596  025304  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
6597  025306  000240                            NOP                             ;RETURN HERE IF ERROR
6598  025310  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
6599  025312  000137  025626                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6600  025316                          80$:
6601
6602                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6603  025316  004737  037432                    JSR     PC,GETSTS
6604
6605                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6606  025322  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6607  025326  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6608  025332  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
6609  025334  000240                            NOP                             ;RETURN HERE IF ERROR
6610  025336  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
6611  025340  000137  025626                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6612  025344                          90$:
6613
6614                                    ;VERIFY RESULTS OF WRITE COMMAND
6615  025344  004737  040512                    JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6616  025350  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
6617  025352  000240                            NOP                             ;RETURN HERE IF ERROR
6618  025354  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6619  025356  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6620  025360  000137  025626                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6621  025364                          100$:
6622  025364  004737  053016                    JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6623  025370  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
6624  025372  000240                            NOP                             ;RETURN HERE IF ERROR
6625  025374  104000                            ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6626  025376  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6627  025400  000137  025626                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6628  025404                          110$:
6629  025404  004737  041344                    JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6630  025410  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
6631  025412  000240                            NOP                             ;RETURN HERE IF ERROR
6632  025414  104000                            ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6633  025416  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6634  025420  000137  025626                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6635  025424                          120$:
6636
6637                                    ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6638  025424  004737  034000                    JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6639  025430  154130                            .WORD   154130  ;TASK DESCRIPTOR
6640  025432  000404                            BR      125$                    ;GO TO 125$ IF NO ERROR
```

```
6641  025434  000240                          NOP                     ;RETURN HERE IF ERROR
6642  025436  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6643  025440  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6644  025444                          125$:
6645
6646                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6647  025444  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
6648  025452  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
6649  025460  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6650  025464  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6651  025466  000240                          NOP                     ;RETURN HERE IF ERROR
6652  025470  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6653  025472  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6654  025476                          130$:
6655
6656                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6657  025476  004737  037432                  JSR     PC,GETSTS
6658
6659                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6660  025502  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6661  025506  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6662  025512  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6663  025514  000240                          NOP                     ;RETURN HERE IF ERROR
6664  025516  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6665  025520  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6666  025524                          140$:
6667
6668                                  ;VERIFY THE RESULTS OF READ OPERATION
6669  025524  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6670  025530  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6671  025532  000240                          NOP                     ;RETURN HERE IF ERROR
6672  025534  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6673  025536  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6674  025540  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6675  025544                          150$:
6676  025544  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6677  025550  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6678  025552  000240                          NOP                     ;RETURN HERE IF ERROR
6679  025554  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6680  025556  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6681  025560  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6682  025564                          160$:
6683  025564  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6684  025570  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6685  025572  000240                          NOP                     ;RETURN HERE IF ERROR
6686  025574  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6687  025576  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6688  025600  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6689  025604                          170$:
6690
6691                                  ;INCREMENT ADDRESS AND FORMAT NEXT CYLINDER
6692  025604  062737  000001  001434          ADD     #1,RMDCO                ;ADVANCE CYLINDER COUNT
6693  025612  022737  001777  001434          CMP     #1777,RMDCO             ;DONE ALL CYLINDERS??
6694  025620  103402                          BLO     190$                    ;YES!!
6695  025622  000137  025142                  JMP     15$                     ;GO DO NEXT SECTOR
6696  025626                          190$:
```

```
6697
6698                                    ;;*****************************************************************
6699                                    ;*TEST 30        FORMAT AT OFFSET
6700                                    ;;*****************************************************************
6701   025626                           TST30:
6702   025626  000004                           SCOPE                   ;SCOPE CALL
6703   025630  000240                           NOP                     ;START OF TEST
6704   025632  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6705   025636  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6706   025642  013701  001450                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6707   025646  012737  000030  001226           MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6708   025654                           10$:
6709
6710                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6711   025654  012737  000000  001434           MOV     #0,RMDC0                ;CYLINDER = 0
6712   025662  012737  000000  001406           MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6713   025670  012737  010000  001432           MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6714   025676  012737  177376  001402           MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6715   025704  012737  103566  001404           MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6716   025712  012737  000063  001400           MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6717   025720  012737  001406  001174           MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6718   025726  012737  000001  001176           MOV     #1,$TMP1
6719   025734  004737  036620           15$:    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6720   025740                           20$:
6721
6722                                    ;PREPARE DEVICE FOR DATA TRANSFER
6723   025740  004737  034000                   JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6724   025744  154130                           .WORD   154130  ;TASK DESCRIPTOR
6725   025746  000404                           BR      30$                     ;GO TO 30$ IF NO ERROR
6726   025750  000240                           NOP                     ;RETURN HERE IF ERROR
6727   025752  104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6728   025754  000137  026520                   JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6729   025760                           30$:
6730
6731                                    ;OFFSET DEVICE FOR WRITE
6732   025760  012737  000015  001400           MOV     #OFFSET!GO,RMCS10       ;CHANGE TO OFFSET COMMAND
6733   025766  012702  001535                   MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
6734   025772  112722  000006                   MOVB    #RMDA,(R2)+
6735   025776  112722  000034                   MOVB    #RMDC,(R2)+
6736   026002  112722  000032                   MOVB    #RMOF,(R2)+
6737   026006  112722  000000                   MOVB    #RMCS1,(R2)+
6738   026012  112712  000200                   MOVB    #200,(R2)               ;TERMINATE TABLE
6739   026016  004737  037766                   JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6740   026022  000404                           BR      35$     ;GO TO 35$ IF NO ERROR
6741   026024  000240                           NOP                     ;RETURN HERE IF ERROR
6742   026026  104000                           ERROR                   ;ERROR DEFINED BY PUT SUB
6743   026030  000137  026520                   JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6744   026034                           35$:
6745
6746                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND AT OFFSET
6747   026034  012737  000063  001400           MOV     #WH!GO,RMCS10           ; WRITE HEADER AND DATA COMMAND
6748   026042  012702  001535                   MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6749   026046  112722  000002                   MOVB    #RMWC,(R2)+
6750   026052  112722  000004                   MOVB    #RMBA,(R2)+
6751   026056  112722  000000                   MOVB    #RMCS1,(R2)+
6752   026062  112722  000200                   MOVB    #200,(R2)+
```

```
6753  026066  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6754  026072  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6755  026074  000240                          NOP                             ;RETURN HERE IF ERROR
6756  026076  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
6757  026100  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6758  026104                          80$:
6759
6760                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6761  026104  004737  037432                  JSR     PC,GETSTS
6762
6763                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6764  026110  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6765  026114  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6766  026120  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6767  026122  000240                          NOP                             ;RETURN HERE IF ERROR
6768  026124  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
6769  026126  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6770  026132                          90$:
6771
6772                                  ;VERIFY RESULTS OF WRITE COMMAND
6773  026132  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6774  026136  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6775  026140  000240                          NOP                             ;RETURN HERE IF ERROR
6776  026142  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6777  026144  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6778  026146  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6779  026152                          100$:
6780  026152  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6781  026156  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
6782  026160  000240                          NOP                             ;RETURN HERE IF ERROR
6783  026162  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6784  026164  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6785  026166  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6786  026172                          110$:
6787  026172  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6788  026176  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6789  026200  000240                          NOP                             ;RETURN HERE IF ERROR
6790  026202  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6791  026204  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6792  026206  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6793  026212                          120$:
6794                                  ;OFFSET DEVICE FOR READ
6795  026212  012737  000015  001400          MOV     #OFFSET!GO,RMCS10       ;CHANGE TO OFFSET COMMAND
6796  026220  012702  001535                  MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
6797  026224  112722  000006                  MOVB    #RMDA,(R2)+
6798  026230  112722  000034                  MOVB    #RMDC,(R2)+
6799  026234  112722  000032                  MOVB    #RMOF,(R2)+
6800  026240  112722  000000                  MOVB    #RMCS1,(R2)+
6801  026244  112722  000200                  MOVB    #200,(R2)+
6802  026250  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6803  026254  000404                          BR      125$    ;GO TO 125$ IF NO ERROR
6804  026256  000240                          NOP                             ;RETURN HERE IF ERROR
6805  026260  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
6806  026262  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6807  026266                          125$:
6808
```

```
 6809
 6810                                   ;READ HEADER AND DATA AT OFFSET
 6811  026266  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
 6812  026274  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
 6813  026302  012702  001535                  MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
 6814  026306  112722  000002                  MOVB    #RMWC,(R2)+
 6815  026312  112722  000004                  MOVB    #RMBA,(R2)+
 6816  026316  112722  000000                  MOVB    #RMCS1,(R2)+
 6817  026322  112722  000200                  MOVB    #200,(R2)+
 6818  026326  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 6819  026332  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
 6820  026334  000240                          NOP                             ;RETURN HERE IF ERROR
 6821  026336  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
 6822  026340  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6823  026344                          130$:
 6824
 6825                                   ;WAIT FOR READ TO COMPLETE AND GET STATUS
 6826  026344  004737  040326                  JSR     PC,TIMOUT                       ;WAIT FOR READ TO COMPLETE
 6827  026350  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 6828  026354  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
 6829  026356  000240                          NOP                             ;RETURN HERE IF ERROR
 6830  026360  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
 6831  026362  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6832  026366                          140$:
 6833
 6834                                   ;VERIFY THE RESULTS OF READ OPERATION
 6835  026366  004737  040512                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
 6836  026372  000405                          BR      150$                            ;GO TO 150$ IF NO ERROR
 6837  026374  000240                          NOP                             ;RETURN HERE IF ERROR
 6838  026376  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
 6839  026400  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 6840  026402  000137  026520                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
 6841  026406                          150$:
 6842  026406  004737  053016                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
 6843  026412  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
 6844  026414  000240                          NOP                             ;RETURN HERE IF ERROR
 6845  026416  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
 6846  026420  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 6847  026422  000137  026520                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
 6848  026426                          160$:
 6849  026426  004737  041344                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
 6850  026432  000405                          BR      170$                            ;GO TO 170$ IF NO ERROR
 6851  026434  000240                          NOP                             ;RETURN HERE IF ERROR
 6852  026436  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
 6853  026440  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 6854  026442  000137  026520                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
 6855  026446                          170$:
 6856
 6857                                   ;VERIFY DATA
 6858  026446  004737  037064                  JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
 6859  026452  103566                          .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
 6860  026454  104572                          .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
 6861  026456  000404                          BR      180$                            ;GO TO 180$ IF NO ERROR
 6862  026460  000240                          NOP                             ;RETURN HERE IF ERROR
 6863  026462  104000                          ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 6864  026464  000137  026520                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
```

```
6865  026470                                  180$:
6866  026470  032737  000200  001432                BIT     #OFD,RMOFO              ;DONE BOTH OFFSETS??
6867  026476  001010                                BNE     190$                    ;YES!!
6868  026500  052737  000200  001432                BIS     #OFD,RMOFO              ;SET FORWARD OFFSET
6869  026506  012737  103566  001404                MOV     #BUFONE,RMBA0           ;CHANGE DATA BUFFER
6870  026514  000137  025740                        JMP     20$
6871  026520                                  190$:
6872
6873                                          ;;*****************************************************************
6874                                          ;*TEST 31        IVC FORMAT TEST
6875                                          ;;*****************************************************************
6876  026520                                  TST31:
6877  026520  000004                                SCOPE                           ;SCOPE CALL
6878  026522  000240                                NOP                             ;START OF TEST
6879  026524  012706  001100                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6880  026530  013700  001276                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6881  026534  013701  001450                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6882  026540  012737  000031  001226                MOV     #31,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6883  026546                                  10$:
6884
6885                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6886  026546  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
6887  026554  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6888  026562  012737  010000  001432                MOV     #FMT16,RMOFO            ;16 BIT FORMAT
6889  026570  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6890  026576  012737  103566  001404                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6891  026604  012737  000062  001400                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6892  026612  012737  001406  001174                MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6893  026620  012737  000001  001176                MOV     #1,$TMP1
6894  026626  004737  036620                  15$:  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6895  026632                                  20$:
6896
6897                                          ;PREPARE DEVICE FOR DATA TRANSFER
6898  026632  004737  034000                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6899  026636  154130                                .WORD   154130  ;TASK DESCRIPTOR
6900  026640  000404                                BR      30$                     ;GO TO 30$ IF NO ERROR
6901  026642  000240                                NOP                     ;RETURN HERE IF ERROR
6902  026644  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6903  026646  000137  027326                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6904  026652                                  30$:
6905
6906                                          ;RESET VOLUME VALID BY SETTING AND RESETTING DIAGNOSTIC MODE
6907  026652  012737  000001  001424                MOV     #DMD,RMMR1O             ;SET DIAGNOSTIC MODE
6908  026660  112737  000024  001535                MOVB    #RMMR1,PUTINX           ;WRITE REGISTER INDEX TABLE
6909  026666  112737  000200  001536                MOVB    #200,PUTINX+1
6910  026674  004737  037766                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6911  026700  000404                                BR      35$     ;GO TO 35$ IF NO ERROR
6912  026702  000240                                NOP                     ;RETURN HERE IF ERROR
6913  026704  104000                                ERROR                   ;ERROR DEFINED BY PUT SUB
6914  026706  000137  027326                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6915  026712                                  35$:
6916
6917                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6918  026712  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6919  026720  012737  000000  001424                MOV     #0,RMMR1O               ;RESET DIAGNOSTIC MODE
6920  026726  012702  001535                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
```

```
6921  026732  112722  000024                  MOVB    #RMMR1,(R2)+
6922  026736  112722  000006                  MOVB    #RMDA,(R2)+
6923  026742  112722  000034                  MOVB    #RMDC,(R2)+
6924  026746  112722  000032                  MOVB    #RMDF,(R2)+
6925  026752  112722  000002                  MOVB    #RMWC,(R2)+
6926  026756  112722  000004                  MOVB    #RMBA,(R2)+
6927  026762  112722  000000                  MOVB    #RMCS1,(R2)+
6928  026766  112722  000200                  MOVB    #200,(R2)+
6929  026772  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6930  026776  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6931  027000  000240                          NOP                     ;RETURN HERE IF ERROR
6932  027002  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6933  027004  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6934  027010                          80$:
6935
6936                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6937  027010  004737  037432                  JSR     PC,GETSTS
6938
6939                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6940  027014  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6941  027020  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6942  027024  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6943  027026  000240                          NOP                     ;RETURN HERE IF ERROR
6944  027030  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6945  027032  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6946  027036                          90$:
6947
6948                                  ;VERIFY RESULTS OF WRITE COMMAND
6949  027036  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6950  027042  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6951  027044  000240                          NOP                     ;RETURN HERE IF ERROR
6952  027046  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6953  027050  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6954  027052  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6955  027056                          100$:
6956  027056  032737  010000  001372          BIT     #IVC,RMER2I             ;IS IVC STATUS SET??
6957  027064  001012                          BNE     110$                    ;YES!!
6958  027066  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
6959  027074  013737  001372  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
6960  027102  052737  010000  001140          BIS     #IVC,$GDDAT
6961  027110  104342                          ERROR   342                     ;IVC NOT SET DURING FORMAT
6962  027112                          110$:
6963  027112  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6964  027116  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6965  027120  000240                          NOP                     ;RETURN HERE IF ERROR
6966  027122  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6967  027124  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6968  027126  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6969  027132                          120$:
6970
6971                                  ;CLEAR IVC ERROR
6972  027132  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6973  027136  040100                          .WORD   040100  ;TASK DESCRIPTOR
6974  027140  000404                          BR      125$                    ;GO TO 125$ IF NO ERROR
6975  027142  000240                          NOP                     ;RETURN HERE IF ERROR
6976  027144  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
6977  027146  000137  027326                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6978  027152                            125$:
6979
6980                                    ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6981  027152  012737  000073  001400              MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
6982  027160  012737  104572  001404              MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
6983  027166  004737  037766                      JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6984  027172  000404                              BR      130$    ;GO TO 130$ IF NO ERROR
6985  027174  000240                              NOP                     ;RETURN HERE IF ERROR
6986  027176  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
6987  027200  000137  027326                      JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6988  027204                            130$:
6989
6990                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6991  027204  004737  037432                      JSR     PC,GETSTS
6992
6993                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
6994  027210  004737  040326                      JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6995  027214  004737  037516                      JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6996  027220  000404                              BR      140$    ;GO TO 140$ IF NO ERROR
6997  027222  000240                              NOP                     ;RETURN HERE IF ERROR
6998  027224  104000                              ERROR                   ;ERROR DEFINED BY GET SUB
6999  027226  000137  027326                      JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
7000  027232                            140$:
7001
7002                                    ;VERIFY THE RESULTS OF READ OPERATION
7003  027232  004737  040512                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7004  027236  000405                              BR      150$                    ;GO TO 150$ IF NO ERROR
7005  027240  000240                              NOP                     ;RETURN HERE IF ERROR
7006  027242  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7007  027244  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7008  027246  000137  027326                      JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
7009  027252                            150$:
7010  027252  032737  010000  001372              BIT     #IVC,RMER2I             ;IS IVC STATUS SET??
7011  027260  001012                              BNE     160$                    ;YES!!
7012  027262  013737  001372  001142              MOV     RMER2I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7013  027270  013737  001372  001140              MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7014  027276  052737  010000  001140              BIS     #IVC,$GDDAT
7015  027304  104342                              ERROR   342                     ;IVC NOT SET DURING FORMAT
7016  027306                            160$:
7017  027306  004737  041344                      JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7018  027312  000405                              BR      170$                    ;GO TO 170$ IF NO ERROR
7019  027314  000240                              NOP                     ;RETURN HERE IF ERROR
7020  027316  104000                              ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7021  027320  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7022  027322  000137  027326                      JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
7023  027326                            170$:
7024  027326                            190$:
7025
7026                                    ;;*****************************************************************
7027                                    ;$TEST 32       FORMAT ERROR TEST - 18
7028                                    ;;*****************************************************************
7029  027326                            TST32:
7030  027326  000004                              SCOPE                   ;SCOPE CALL
7031  027330  000240                              NOP                     ;START OF TEST
7032  027332  012706  001100                      MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
 7033  027336  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 7034  027342  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 7035  027346  012737  000032  001226          MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 7036  027354                                10$:
 7037
 7038                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 7039  027354  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
 7040  027362  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 7041  027370  012737  000000  001432          MOV     #0,RMOF0                ;18 BIT FORMAT
 7042  027376  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 7043  027404  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 7044  027412  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 7045  027420  012737  065106  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
 7046  027426  012737  000001  001176          MOV     #1,$TMP1
 7047  027434  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 7048  027440                                20$:
 7049
 7050                                        ;PREPARE DEVICE FOR DATA TRANSFER
 7051  027440  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7052  027444  154130                          .WORD   154130  ;TASK DESCRIPTOR
 7053  027446  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
 7054  027450  000240                          NOP                     ;RETURN HERE IF ERROR
 7055  027452  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7056  027454  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7057  027460                                30$:
 7058
 7059                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
 7060  027460  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
 7061  027466  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 7062  027472  112722  000006                  MOVB    #RMDA,(R2)+
 7063  027476  112722  000034                  MOVB    #RMDC,(R2)+
 7064  027502  112722  000032                  MOVB    #RMOF,(R2)+
 7065  027506  112722  000002                  MOVB    #RMWC,(R2)+
 7066  027512  112722  000004                  MOVB    #RMBA,(R2)+
 7067  027516  112722  000000                  MOVB    #RMCS1,(R2)+
 7068  027522  112722  000200                  MOVB    #200,(R2)+
 7069  027526  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 7070  027532  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
 7071  027534  000240                          NOP                     ;RETURN HERE IF ERROR
 7072  027536  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 7073  027540  000137  030050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7074  027544                                80$:
 7075
 7076                                        ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
 7077  027544  004737  037432                  JSR     PC,GETSTS
 7078
 7079                                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 7080  027550  004737  040326                  JSR     PC,TIMOUT                       ;WAIT FOR COMMAND TO COMPLETE
 7081  027554  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 7082  027560  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
 7083  027562  000240                          NOP                     ;RETURN HERE IF ERROR
 7084  027564  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 7085  027566  000137  030050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7086  027572                                90$:
 7087
 7088                                        ;VERIFY RESULTS OF WRITE COMMAND
```

```
7089  027572  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7090  027576  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
7091  027600  000240                  NOP                     ;RETURN HERE IF ERROR
7092  027602  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7093  027604  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7094  027606  000137  030050          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7095  027612                  100$:
7096  027612  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7097  027616  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
7098  027620  000240                  NOP                     ;RETURN HERE IF ERROR
7099  027622  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7100  027624  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7101  027626  000137  030050          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7102  027632                  110$:
7103  027632  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7104  027636  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
7105  027640  000240                  NOP                     ;RETURN HERE IF ERROR
7106  027642  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7107  027644  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7108  027646  000137  030050          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7109  027652                  120$:
7110
7111
7112                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 16 BIT FORMAT
7113  027652  012737  010000  001432          MOV     #FMT16.RMOFO            ;CHANGE TO 16 BIT FORMAT
7114  027660  012737  000073  001400          MOV     #RH!GO,RMCSIO           ;READ HEADER & DATA COMMAND
7115  027666  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
7116  027674  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7117  027700  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
7118  027702  000240                  NOP                     ;RETURN HERE IF ERROR
7119  027704  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7120  027706  000137  030050          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7121  027712                  130$:
7122
7123                          ;WAIT FOR READ TO COMPLETE AND GET STATUS
7124  027712  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7125  027716  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7126  027722  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
7127  027724  000240                  NOP                     ;RETURN HERE IF ERROR
7128  027726  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
7129  027730  000137  030050          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7130  027734                  140$:
7131
7132                          ;VERIFY THE RESULTS OF READ OPERATION
7133  027734  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7134  027740  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
7135  027742  000240                  NOP                     ;RETURN HERE IF ERROR
7136  027744  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7137  027746  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7138  027750  000137  030050          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7139  027754                  150$:
7140
7141                          ;VERIFY THAT FORMAT ERROR IS SET
7142  027754  032737  000020  001344          BIT     #FER,RMER1I             ;IS FORMAT ERROR SET??
7143  027762  001022                  BNE     160$                    ;YES!!
7144  027764  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
```

```
7145  027770  000405                          BR      155$                    ;GO TO 155$ IF NO ERROR
7146  027772  000240                          NOP                     ;RETURN HERE IF ERROR
7147  027774  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7148  027776  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7149  030000  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7150  030004  013737  001344  001142  155$:   MOV     RMER11,$BDDAT           ;BAD DATA FOR TYPEOUT
7151  030012  013737  001344  001140          MOV     RMER11,$GDDAT           ;EXPECTED DATA
7152  030020  052737  000020  001140          BIS     #FER,$GDDAT
7153  030026  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7154  030030                          160$:
7155  030030  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7156  030034  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
7157  030036  000240                          NOP                     ;RETURN HERE IF ERROR
7158  030040  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7159  030042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7160  030044  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7161  030050                          170$:
7162  030050                          180$:
7163
7164                                  ;;*****************************************************************
7165                                  ;*TEST 33       FORMAT ERROR TEST - 16
7166                                  ;;*****************************************************************
7167  030050                          TST33:
7168  030050  000004                          SCOPE                   ;SCOPE CALL
7169  030052  000240                          NOP                     ;START OF TEST
7170  030054  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7171  030060  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7172  030064  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7173  030070  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7174  030076                          10$:
7175
7176                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7177  030076  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
7178  030104  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7179  030112  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7180  030120  012737  177376  001402          MOV     #<^C<2*256.>+1>,RMWC0   ;2 * 256 WORDS
7181  030126  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7182  030134  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7183  030142  012737  065106  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7184  030150  012737  000001  001176          MOV     #1,$TMP1
7185  030156  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7186  030162                          20$:
7187
7188                                  ;PREPARE DEVICE FOR DATA TRANSFER
7189  030162  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7190  030166  154130                          .WORD   154!30  ;TASK DESCRIPTOR
7191  030170  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
7192  030172  000240                          NOP                     ;RETURN HERE IF ERROR
7193  030174  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7194  030176  000137  030572                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7195  030202                          30$:
7196
7197                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7198  030202  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7199  030210  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7200  030214  112722  000006                  MOVB    #RMDA,(R2)+
```

```
7201  030220  112722  000034          MOVB    #RMDC,(R2)+
7202  030224  112722  000032          MOVB    #RMOF,(R2)+
7203  030230  112722  000002          MOVB    #RMWC,(R2)+
7204  030234  112722  000004          MOVB    #RMBA,(R2)+
7205  030240  112722  000000          MOVB    #RMCS1,(R2)+
7206  030244  112722  000200          MOVB    #200,(R2)+
7207  030250  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7208  030254  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
7209  030256  000240                  NOP                     ;RETURN HERE IF ERROR
7210  030260  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7211  030262  000137  030572          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7212  030266                  80$:
7213
7214                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7215  030266  004737  037432          JSR     PC,GETSTS
7216
7217                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7218  030272  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7219  030276  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7220  030302  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
7221  030304  000240                  NOP                     ;RETURN HERE IF ERROR
7222  030306  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
7223  030310  000137  030572          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7224  030314                  90$:
7225
7226                          ;VERIFY RESULTS OF WRITE COMMAND
7227  030314  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7228  030320  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
7229  030322  000240                  NOP                     ;RETURN HERE IF ERROR
7230  030324  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7231  030326  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7232  030330  000137  030572          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7233  030334                  100$:
7234  030334  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7235  030340  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
7236  030342  000240                  NOP                     ;RETURN HERE IF ERROR
7237  030344  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7238  030346  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7239  030350  000137  030572          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7240  030354                  110$:
7241  030354  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7242  030360  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
7243  030362  000240                  NOP                     ;RETURN HERE IF ERROR
7244  030364  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7245  030366  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7246  030370  000137  030572          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7247  030374                  120$:
7248
7249
7250                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 18 BIT FORMAT
7251  030374  012737  000000  001432          MOV     #0,RMOF0                ;SET FOR 18 BIT FORMAT
7252  030402  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
7253  030410  012737  104572  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
7254  030416  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7255  030422  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7256  030424  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7257  030426  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7258  030430  000137  030572                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7259  030434                        130$:
7260
7261                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
7262  030434  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7263  030440  004737  037516                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
7264  030444  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
7265  030446  000240                        NOP                     ;RETURN HERE IF ERROR
7266  030450  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7267  030452  000137  030572                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7268  030456                        140$:
7269
7270                                  ;VERIFY THE RESULTS OF READ OPERATION
7271  030456  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7272  030462  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7273  030464  000240                        NOP                     ;RETURN HERE IF ERROR
7274  030466  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7275  030470  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7276  030472  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7277  030476                        150$:
7278
7279                                  ;VERIFY THAT FORMAT ERROR IS SET
7280  030476  032737  000020  001344        BIT     #FER,RMER1I             ;IS FORMAT ERROR SET??
7281  030504  001022                        BNE     160$                    ;YES!!
7282  030506  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7283  030512  000405                        BR      155$                    ;GO TO 155$ IF NO ERROR
7284  030514  000240                        NOP                     ;RETURN HERE IF ERROR
7285  030516  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7286  030520  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7287  030522  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7288  030526  013737  001344  001142  155$:  MOV     RMER1I,$BNDAT           ;RECEIVED STATUS FOR TYPEOUT
7289  030534  013737  001344  001140        MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7290  030542  052737  000020  001140        BIS     #FER,$GDDAT
7291  030550  104343                        ERROR   343                     ;FORMAT ERROR NOT SET
7292  030552                        160$:
7293  030552  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7294  030556  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
7295  030560  000240                        NOP                     ;RETURN HERE IF ERROR
7296  030562  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7297  030564  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7298  030566  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7299  030572                        170$:
7300  030572                        180$:
7301
7302                                  ;;*****************************************************************
7303                                  ;*TEST 34        FORMAT HCE, FIRST HEADER WORD
7304                                  ;;*****************************************************************
7305  030572                        TST34:
7306  030572  000004                        SCOPE                   ;SCOPE CALL
7307  030574  000240                        NOP                     ;START OF TEST
7308  030576  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7309  030602  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7310  030606  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7311  030612  012737  000034  001226        MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7312  030620  012737  000001  031576        MOV     #1,190$                 ;INIT BIT POSITION
```

```
7313  030626                                  10$:
7314
7315                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7316  030626  012737  000000  001434                MOV     #0,RMOC0                ;CYLINDER = 0
7317  030634  012737  000000  001406                MOV     #0,RMOA0                ;TRACK = SECTOR = 0
7318  030642  012737  010000  001432                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7319  030650  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7320  030656  012737  103566  001404                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7321  030664  012737  000062  001400                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7322  030672  012737  065106  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7323  030700  012737  000001  001176                MOV     #1,$TMP1
7324  030706  004737  036620                      JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7325  030712                                  20$:
7326
7327                                          ;PREPARE DEVICE FOR DATA TRANSFER
7328  030712  004737  034000                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7329  030716  154130                              .WORD   154130  ;TASK DESCRIPTOR
7330  030720  000404                              BR      30$                     ;GO TO 30$ IF NO ERROR
7331  030722  000240                              NOP                     ;RETURN HERE IF ERROR
7332  030724  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7333  030726  000137  031750                      JMP     240$                    ;GO TO 240$ IF ERROR WAS FOUND
7334  030732                                  30$:
7335
7336                                          ;COMPLEMENT DATA BIT IN FIRST HEADER WORD
7337  030732  033737  031576  103566              BIT     #190$,BUFONE            ;IS BIT IN HEADER ON??
7338  030740  001404                              BEQ     31$                     ;NO!!
7339  030742  043737  031576  103566              BIC     #190$,BUFONE            ;RESET BIT IN HEADER
7340  030750  000403                              BR      32$
7341  030752  053737  031576  103566  31$:        BIS     #190$,BUFONE            ;SET BIT IN HEADER
7342
7343  030760                                  32$:
7344
7345                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7346  030760  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7347  030766  012702  001535                      MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7348  030772  112722  000006                      MOVB    #RMOA,(R2)+
7349  030776  112722  000034                      MOVB    #RMOC,(R2)+
7350  031002  112722  000032                      MOVB    #RMOF,(R2)+
7351  031006  112722  000002                      MOVB    #RMWC,(R2)+
7352  031012  112722  000004                      MOVB    #RMBA,(R2)+
7353  031016  112722  000000                      MOVB    #RMCS1,(R2)+
7354  031022  112722  000200                      MOVB    #200,(R2)+
7355  031026  004737  037766                      JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
7356  031032  000404                              BR      80$     ;GO TO 80$ IF NO ERROR
7357  031034  000240                              NOP                     ;RETURN HERE IF ERROR
7358  031036  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
7359  031040  000137  031574                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7360  031044                                  80$.
7361
7362                                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7363  031044  004737  037432                      JSR     PC,GETSTS
7364
7365                                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7366  031050  004737  040326                      JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7367  031054  004737  037516                      JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
7368  031060  000404                              BR      90$     ;GO TO 90$ IF NO ERROR
```

```
 7369  031062  000240                        NOP                     ;RETURN HERE IF ERROR
 7370  031064  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
 7371  031066  000137  031574                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7372  031072                        90$:
 7373
 7374                                ;VERIFY RESULTS OF WRITE COMMAND
 7375  031072  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7376  031076  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
 7377  031100  000240                        NOP                     ;RETURN HERE IF ERROR
 7378  031102  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7379  031104  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7380  031106  000137  031574                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7381  031112                        100$:
 7382  031112  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 7383  031116  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
 7384  031120  000240                        NOP                     ;RETURN HERE IF ERROR
 7385  031122  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 7386  031124  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7387  031126  000137  031574                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7388  031132                        110$:
 7389  031132  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 7390  031136  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
 7391  031140  000240                        NOP                     ;RETURN HERE IF ERROR
 7392  031142  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 7393  031144  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7394  031146  000137  031574                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7395  031152                        120$:
 7396
 7397
 7398                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
 7399  031152  012737  000073  001400        MOV     #RH!GO,RMCS1O                   ;READ HEADER & DATA COMMAND
 7400  031160  012737  104572  001404        MOV     #BUFTWO,RMBAO                   ;CHANGE BUS ADDRESS
 7401  031166  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7402  031172  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
 7403  031174  000240                        NOP                     ;RETURN HERE IF ERROR
 7404  031176  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
 7405  031200  000137  031574                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7406  031204                        130$:
 7407
 7408                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
 7409  031204  004737  040326                JSR     PC,TIMOUT                       ;WAIT FOR READ TO COMPLETE
 7410  031210  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7411  031214  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
 7412  031216  000240                        NOP                     ;RETURN HERE IF ERROR
 7413  031220  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
 7414  031222  000137  031574                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7415  031226                        140$:
 7416
 7417                                ;VERIFY THE RESULTS OF READ OPERATION
 7418  031226  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7419  031232  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
 7420  031234  000240                        NOP                     ;RETURN HERE IF ERROR
 7421  031236  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7422  031240  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7423  031242  000137  031574                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7424  031246                        150$:
```

```
7425  031246  032737  140000  031576          BIT     #MSE!USE,190$           ;IS THIS BSE ??
7426  031254  001033                          BNE     152$                    ;YES !!
7427  031256  032737  010000  031576          BIT     #FMT16,190$             ;IS THIS FER ??
7428  031264  001466                          BEQ     155$                    ;NO !!
7429
7430                                  ;VERIFY THAT FER IS SET
7431  031266  032737  000020  001344          BIT     #FER,RMER1I             ;IS FER SET ??
7432  031274  001122                          BNE     160$                    ;YES !!
7433  031276  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7434  031302  000405                          BR      151$                    ;GO TO 151$ IF NO ERROR
7435  031304  000240                          NOP                     ;RETURN HERE IF ERROR
7436  031306  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7437  031310  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7438  031312  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7439  031316  013737  001344  001142  151$:   MOV     RMER1I,$RDDAT           ;RECEIVED STATUS
7440  031324  013737  001344  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7441  031332  052737  000020  001140          BIS     #FER,$GDDAT
7442  031340  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7443  031342  000514                          BR      180$
7444  031344                          152$:
7445                                  ;VERIFY THAT BAD SECTOR ERROR IS SET
7446  031344  032737  100000  001372          BIT     #BSE,RMER2I             ;IS BSE SET ??
7447  031352  001073                          BNE     160$                    ;YES !!
7448  031354  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7449  031360  000405                          BR      153$                    ;GO TO 153$ IF NO ERROR
7450  031362  000240                          NOP                     ;RETURN HERE IF ERROR
7451  031364  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7452  031366  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7453  031370  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7454  031374                          153$:
7455  031374  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
7456  031402  013737  001372  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7457  031410  052737  100000  001140          BIS     #BSE,$GDDAT
7458  031416  013737  103566  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7459  031424  013737  104572  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7460  031432  013737  031576  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7461                                  ;       ERROR   345                     ;FAILED TO DETECT BSE
7462  031440  000455                          BR      180$                    ;SKIP REST OF TEST
7463  031442                          155$:
7464
7465                                  ;VERIFY THAT HEADER COMPARE ERROR IS SET
7466  031442  032737  000200  001344          BIT     #HCE,RMER1I             ;IS "HCE" SET??
7467  031450  001034                          BNE     160$                    ;YES!!
7468  031452  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7469  031456  000405                          BR      156$                    ;GO TO 156$ IF NO ERROR
7470  031460  000240                          NOP                     ;RETURN HERE IF ERROR
7471  031462  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7472  031464  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7473  031466  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7474  031472  013737  001344  001142  156$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7475  031500  013737  001344  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7476  031506  052737  000200  001140          BIS     #HCE,$GDDAT
7477  031514  013737  103566  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7478  031522  013737  104572  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7479  031530  013737  031576  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7480  031536  104344                          ERROR   344                     ;FORMAT ERROR NOT SET
```

```
 7481  031540  000415                          BR      180$
 7482
 7483  031542                          160$:
 7484
 7485                                  ;CHECK FOR OTHER ERRORS
 7486  031542  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 7487  031546  000405                          BR      165$                    ;GO TO 165$ IF NO ERROR
 7488  031550  000240                          NOP                     ;RETURN HERE IF ERROR
 7489  031552  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 7490  031554  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7491  031556  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7492  031562                          165$:
 7493
 7494
 7495
 7496                                  ;ADVANCE THE BIT POSITION AND FORMAT AGAIN IF NOT DONE
 7497  031562  006337  031576                  ASL     190$                    ;SHIFT TO NEXT BIT POSITION
 7498  031566  001404                          BEQ     200$                    ;EXIT IF DONE
 7499  031570  000137  030626                  JMP     10$                     ;GO DO NEXT SECTOR
 7500  031574                          180$:
 7501  031574  000465                          BR      240$                    ;JUMP OVER STORAGE
 7502  031576                          190$:
 7503  031576  000000                          .WORD                           ;STORAGE FOR BIT POSITION
 7504  031600                          200$:
 7505                                  ;       MOV     #0,RMDC0                ;REFORMAT THE SECTOR IN 16 BIT MODE
 7506                                  ;       MOV     #0,RMDA0                ;
 7507  031600  012737  010000  001432          MOV     #FMT16,RMOF0            ; 16 BIT MODE
 7508  031606  012737  177776  001402          MOV     #-2,RMWC0               ;ONLY TWO HEADER WORDS
 7509  031614  012737  103566  001404          MOV     #BUFONE,RMBA0           ;SELECT THE BUFFER
 7510  031622  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEAD AND DATA COMMAND
 7511  031630  004737  036620                  JSR     PC,GENBUF               ;GENERATE THE BUFFER PATTERN
 7512
 7513  031634  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7514  031640  154130                          .WORD   154130  ;TASK DESCRIPTOR
 7515  031642  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
 7516  031644  000240                          NOP                     ;RETURN HERE IF ERROR
 7517  031646  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7518  031650  000137  031654                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
 7519  031654  012737  000063  001400  210$:   MOV     #WH!GO,RMCS10           ;SET THE GO BIT
 7520  031662  012702  001535                  MOV     #PUTINX,R2              ;PUT THE REGISTER ADDRESS INTO TABLE
 7521  031666  112722  000006                  MOVB    #RMDA,(R2)+             ;
 7522  031672  112722  000034                  MOVB    #RMDC,(R2)+             ;
 7523  031676  112722  000032                  MOVB    #RMOF,(R2)+             ;
 7524  031702  112722  000002                  MOVB    #RMWC,(R2)+             ;
 7525  031706  112722  000004                  MOVB    #RMBA,(R2)+             ;
 7526  031712  112722  000000                  MOVB    #RMCS1,(R2)+            ;
 7527  031716  112722  000200                  MOVB    #200,(R2)+              ;TERMINATOR
 7528  031722  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 7529  031726  000404                          BR      220$    ;GO TO 220$ IF NO ERROR
 7530  031730  000240                          NOP                             ;RETURN HERE IF ERROR
 7531  031732  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
 7532  031734  000137  031750                  JMP     240$    ;GO TO 240$ IF ERROR WAS FOUND
 7533  031740  000240                  220$:   NOP
 7534  031742  004737  040326                  JSR     PC,TIMOUT               ;WAOT UNTIL IT FINISH
 7535  031746  000240                          NOP
 7536  031750                          240$:
```

```
                                ;;*******************************************************************
                                ;*TEST 35        FORMAT HCE, SECOND HEADER WORD
                                ;;*******************************************************************
031750                          T$T35:
031750  000004                          SCOPE                   ;SCOPE CALL
031752  000240                          NOP                     ;START OF TEST
031754  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
031760  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
031764  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
031770  012737  000035  001226          MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
031776  012737  000001  032560          MOV     #1,190$                 ;INIT BIT POSITION
032004                          10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
032004  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
032012  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
032020  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
032026  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
032034  012737  103566  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
032042  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
032050  012737  065106  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
032056  012737  000001  001176          MOV     #1,$TMP1
032064  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
032070                          20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
032070  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
032074  154130                          .WORD   154130  ;TASK DESCRIPTOR
032076  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
032100  000240                          NOP                     ;RETURN HERE IF ERROR
032102  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
032104  000137  032732                  JMP     240$                    ;GO TO 240$ IF ERROR WAS FOUND
032110                          30$:

                                ;COMPLEMENT DATA BIT IN SECOND HEADER WORD
032110  033737  032560  103570          BIT     190$,BUFONE+2           ;IS BIT IN HEADER ON??
032116  001404                          BEQ     31$                     ;NO!!
032120  043737  032560  103570          BIC     190$,BUFONE+2           ;RESET BIT IN HEADER
032126  000403                          BR      32$
032130  053737  032560  103570  31$:    BIS     190$,BUFONE+2           ;SET BIT IN HEADER

032136                          32$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
032136  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
032144  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
032150  112722  000006                  MOVB    #RMDA,(R2)+
032154  112722  000034                  MOVB    #RMDC,(R2)+
032160  112722  000032                  MOVB    #RMOF,(R2)+
032164  112722  000002                  MOVB    #RMWC,(R2)+
032170  112722  000004                  MOVB    #RMBA,(R2)+
032174  112722  000000                  MOVB    #RMCS1,(R2)+
032200  112722  000200                  MOVB    #200,(R2)+
032204  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
032210  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
```

```
7593  032212  000240                        NOP                     ;RETURN HERE IF ERROR
7594  032214  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7595  032216  000137  032556                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7596  032222                        80$:
7597
7598                                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7599  032222  004737  037432                JSR     PC,GETSTS
7600
7601                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7602  032226  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7603  032232  004737  037516                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
7604  032236  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7605  032240  000240                        NOP                             ;RETURN HERE IF ERROR
7606  032242  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
7607  032244  000137  032556                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7608  032250                        90$:
7609
7610                                ;VERIFY RESULTS OF WRITE COMMAND
7611  032250  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7612  032254  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
7613  032256  000240                        NOP                             ;RETURN HERE IF ERROR
7614  032260  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
7615  032262  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7616  032264  000137  032556                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7617  032270                        100$:
7618  032270  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7619  032274  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
7620  032276  000240                        NOP                             ;RETURN HERE IF ERROR
7621  032300  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
7622  032302  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7623  032304  000137  032556                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7624  032310                        110$:
7625  032310  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7626  032314  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
7627  032316  000240                        NOP                             ;RETURN HERE IF ERROR
7628  032320  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
7629  032322  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7630  032324  000137  032556                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7631  032330                        120$:
7632
7633
7634                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7635  032330  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
7636  032336  012737  104572  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
7637  032344  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7638  032350  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
7639  032352  000240                        NOP                             ;RETURN HERE IF ERROR
7640  032354  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
7641  032356  000137  032556                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7642  032362                        130$:
7643
7644                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
7645  032362  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7646  032366  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7647  032372  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
7648  032374  000240                        NOP                             ;RETURN HERE IF ERROR
```

```
7649  032376  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7650  032400  000137  032556                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7651  032404                          140$:
7652
7653                                  ;VERIFY THE RESULTS OF READ OPERATION
7654  032404  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7655  032410  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7656  032412  000240                          NOP                     ;RETURN HERE IF ERROR
7657  032414  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7658  032416  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7659  032420  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7660  032424                          150$:
7661
7662                                  ;VERIFY THAT HEADER COMPARE ERROR IS SET
7663  032424  032737  000200  001344          BIT     #HCE,RMER1I             ;IS "HCE" SET??
7664  032432  001034                          BNE     160$                    ;YES!!
7665  032434  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7666  032440  000405                          BR      155$                    ;GO TO 155$ IF NO ERROR
7667  032442  000240                          NOP                     ;RETURN HERE IF ERROR
7668  032444  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7669  032446  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7670  032450  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7671  032454  013737  001344  001142  155$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7672  032462  013737  001344  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7673  032470  052737  000200  001140          BIS     #HCE,$GDDAT
7674  032476  013737  103570  001174          MOV     BUFONE+2,$TMP0          ;WRITTEN HEADER
7675  032504  013737  104574  001176          MOV     BUFTWO+2,$TMP1          ;READ HEADER WORD
7676  032512  013737  032560  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7677  032520  104344                          ERROR   344                     ;FORMAT ERROR NOT SET
7678  032522  000415                          BR      180$
7679
7680  032524                          160$:
7681
7682                                  ;CHECK FOR OTHER ERRORS
7683  032524  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7684  032530  000405                          BR      165$                    ;GO TO 165$ IF NO ERROR
7685  032532  000240                          NOP                     ;RETURN HERE IF ERROR
7686  032534  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7687  032536  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7688  032540  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7689  032544                          165$:
7690
7691
7692                                  ;ADVANCE THE BIT POSITION AND FORMAT NEXT SECTOR IF NOT DONE
7693  032544  006337  032560                  ASL     190$                    ;SHIFT TO NEXT BIT POSITION
7694  032550  001404                          BEQ     200$                    ;EXIT IF DONE
7695  032552  000137  032004                  JMP     10$                     ;GO DO NEXT SECTOR
7696  032556                          180$:
7697  032556  000465                          BR      240$                    ;JUMP OVER STORAGE
7698  032560                          190$:
7699  032560  000000                          .WORD                   ;STORAGE FOR BIT POSITION
7700  032562                          200$:
7701                                  ;       MOV     #0,RMDC0                ;REFORMAT THE SECTOR IN 16 BIT MODE
7702                                  ;       MOV     #0,RMDA0                ;SECTOR 0,TRACK 0, CYL 0
7703  032562  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT MODE
7704  032570  012737  177776  001402          MOV     #-2,RMWC0               ;2 HEADER WORDS ONLY
```

```
 7705  032576  012737  103566  001404          MOV     #BUFONE,RMBAO           ;BUFFER ADDRESS
 7706  032604  012737  000062  001400          MOV     #WH,RMCS1O              ;WRITE HEAD AND DATA COMMAND
 7707  032612  004737  036620                  JSR     PC,GENBUF               ;SET UP THE BUFFER
 7708  032616  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7709  032622  154130                          .WORD   154130  ;TASK DESCRIPTOR
 7710  032624  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
 7711  032626  000240                          NOP                     ;RETURN HERE IF ERROR
 7712  032630  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7713  032632  000137  032636                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
 7714  032636  012737  000063  001400  210$:   MOV     #WH!GO,RMCS1O           ;SET GO BIT
 7715  032644  012702  001535                  MOV     #PUTINX,R2              ;TABLE ADDRESS
 7716  032650  112722  000006                  MOVB    #RMDA,(R2)+             ;
 7717  032654  112722  000034                  MOVB    #RMOC,(R2)+             ;
 7718  032660  112722  000032                  MOVB    #RMOF,(R2)+             ;
 7719  032664  112722  000004                  MOVB    #RMBA,(R2)+             ;
 7720  032670  112722  000002                  MOVB    #RMWC,(R2)+             ;
 7721  032674  112722  000000                  MOVB    #RMCS1,(R2)+            ;
 7722  032700  112722  000200                  MOVB    #200,(R2)+              ;TERMINATOR
 7723  032704  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7724  032710  000404                          BR      220$    ;GO TO 220$ IF NO ERROR
 7725  032712  000240                          NOP                     ;RETURN HERE IF ERROR
 7726  032714  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 7727  032716  000137  032732                  JMP     240$    ;GO TO 240$ IF ERROR WAS FOUND
 7728  032722  000240          220$:   NOP
 7729  032724  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR TIME OUT
 7730  032730  000240                          NOP
 7731  032732                  240$:
 7732
 7733                          ;PUT NEWTEST HERE
```

```
7734  032732                          $EOSP:
7735  032732  000240                          NOP
7736  032734  013700  001450                  MOV     TSTQUE,R0
7737  032740  062700  000002                  ADD     #2,R0
7738  032744  010037  001450                  MOV     R0,TSTQUE
7739  032750  005710                          TST     (R0)
7740  032752  001402                          BEQ     1$
7741  032754  000137  007124                  JMP     READY
7742  032760  012737  001452  001450  1$:     MOV     #TSTQUE+2,TSTQUE
7743                           .SBTTL  END OF PASS ROUTINE
7744
7745                           ;;**********************************************************************
7746                           ;*INCREMENT THE PASS NUMBER ($PASS)
7747                           ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7748                           ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7749                           ;*IF THERES A MONITOR GO TO IT
7750                           ;*IF THERE ISN'T JUMP TO SHUT
7751
7752  032766                          $EOP:
7753  032766  000240                          NOP
7754  032770  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
7755  032774  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
7756  033000  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
7757  033004  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
7758  033012  005327                          DEC     (PC)+           ;;LOOP?
7759  033014  000001                  $EOPCT: .WORD   1
7760  033016  003063                          BGT     $DOAGN          ;;YES
7761  033020  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
7762  033022  000001                  $ENDCT: .WORD   1
7763  033024  033014                          $EOPCT
7764  033026  104401  033034                  TYPE    ,65$            ;;TYPE ASCIZ STRING
7765  033032  000407                          BR      64$             ;;GET OVER THE ASCIZ
7766                           ;;65$:  .ASCIZ  <12><15>/END PASS #/
7767  033052                  64$:
7768  033052  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
7769                                                                   ;;TYPE PASS NUMBER
7770  033056  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7771  033060  104401  033066                  TYPE    ,67$            ;;TYPE ASCIZ STRING
7772  033064  000421                          BR      66$             ;;GET OVER THE ASCIZ
7773                           ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
7774  033130                  66$:
7775  033130  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
7776                                                                   ;;TOTAL NUMBER OF ERRORS
7777  033134  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7778  033136  104401  001217                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
7779  033142  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
7780  033146  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
7781  033152  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
7782  033154  000005                          RESET                   ;;CLEAR THE WORLD
7783  033156  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
7784  033160  000240                          NOP                     ;;SAVE ROOM
7785  033162  000240                          NOP                     ;;FOR
7786  033164  000240                          NOP                     ;;ACT11
7787  033166                          $DOAGN:
7788  033166  000137                          JMP     @(PC)+          ;;RETURN
7789  033170  057364                  $RTNAD: .WORD   SHUT
```

```
 7790  033172     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 7791  033176             .EVEN
```

```
                                        .SBTTL  SUBROUTINES

                                        ;;******************************************************************
                                        .SBTTL  ERROR TYPEOUT ROUTINE

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

033176                                  ERRTYP:
033176  104414                                  SAVREG
033200  032777  020000  145746                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
033206  001402                                  BEQ     1$              ;NO!!
033210  000137  033726                          JMP     2$              ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
033214  104401  001217                1$:       TYPE    ,$CRLF
033220  104401  033742                          TYPE    ,ERTY00         ;TYPE "UNT#"
033224  013746  001234                          MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
033230  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
033232     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
033233     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
033234  005037  033732                          CLR     TSTNMB          ;LOAD TEST NUMBER FOR
033240  013737  001226  033732                  MOV     $TESTN,TSTNMB
033246  104401  033750                          TYPE    ,ERTY01         ;TYPE "TST#"
033252  013746  033732                          MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
033256  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
033260     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
033261     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
033262  005037  033734                          CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
033266  113737  001130  033734                  MOVB    $ITEMB,ERRNMB   ;TYPEOUT
033274  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
033276  104401  033760                          TYPE    ,ERTY02         ;TYPE "ERR#"
033302  013746  033734                          MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
033306  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
033310     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
033311     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
033312  104401  033767                2$:       TYPE    ,ERTY03         ;TYPE "PC="
033316  013746  001132                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
033322  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
033324     006                                  .BYTE   6               ;;TYPE 6 DIGIT(S)
033325     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
033326  005737  033734                3$:       TST     ERRNMB          ;WAS AN ERROR CALLED?
033332  001575                                  BEQ     21$             ;NO!!
033334  104401  001217                          TYPE    ,$CRLF          ;YES-TYPE CRLF
033340  105037  033740                          CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
7848  033344  105037  033741            CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
7849  033350  013700  033734            MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
7850  033354  006300                    ASL     R0              ;FOUR ENTRIES IN ERROR
7851  033356  006300                    ASL     R0              ;TABLE
7852  033360  006300                    ASL     R0
7853  033362  062700  001554            ADD     #$ERRTB-8.,R0
7854  033366  011001                    MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
7855                                                            ;TABLE
7856  033370  001507                    BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
7857                          ;TYPE THE ERROR MESSAGE
7858  033372  012102            4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
7859  033374  001505                    BEQ     12$             ;BRANCH IF END OF MESSAGE
7860  033376  010237  033544            MOV     R2,11$          ;LOAD ADDRESS OF STRING
7861  033402  005037  033736            CLR     BOTADR          ;CLEAR BOT ADDRESS
7862  033406  112203            5$:     MOVB    (R2)+,R3        ;END OF STRING??
7863  033410  001454                    BEQ     10$             ;YES!!
7864  033412  122703  000015            CMPB    #CR,R3          ;CARRIAGE RETURN??
7865  033416  001003                    BNE     6$              ;NO!!
7866  033420  105037  033741            CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
7867  033424  000770                    BR      5$              ;GET NEXT CHARACTER
7868  033426  122703  000012            CMPB    #LF,R3          ;LINE FEED??
7869  033432  001765                    BEQ     5$              ;YES-GET NEXT CHARACTER
7870  033434  122703  000011            CMPB    #HT,R3          ;HORIZONTAL TAB??
7871  033440  001007                    BNE     8$              ;NO!!
7872  033442  105237  033741            INCB    CHRCNT          ;ADJUST CHARACTER COUNT
7873  033446  132737  000007  033741    BITB    #7,CHRCNT
7874  033454  001372                    BNE     7$
7875  033456  000407                    BR      9$
7876  033460  105237  033741            INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
7877  033464  122703  000040            CMPB    #' ,R3          ;SPACE??
7878  033470  001002                    BNE     9$              ;NO!!
7879  033472  010237  033736            MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
7880  033476  122737  000100  033741    CMPB    #64.,CHRCNT     ;END OF LINE??
7881  033504  103340                    BHIS    5$              ;NO!!
7882  033506  013704  033736            MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
7883  033512  001007                    BNE     90$             ;BRANCH IF SPACE DETECTED
7884  033514  104401  001217            TYPE    ,$CRLF          ;TYPE CRLF
7885  033520  105037  033741            CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7886  033524  013702  033544            MOV     11$,R2          ;SET UP R2 FOR TESTING
7887  033530  000726                    BR      5$
7888  033532  105044            90$:    CLRB    -(R4)           ;REPLACE SPACE
7889  033534  112737  177777  033740    MOVB    #-1,BOTFLG      ;SET BOT FLAG
7890  033542  104401            10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
7891  033544  000000            11$:    .WORD                   ;STRING ADDRESS GOES HERE
7892  033546  105737  033740            TSTB    BOTFLG          ;WAS STRING TRUNCATED??
7893  033552  001707                    BEQ     4$              ;NO!!
7894  033554  104401  001217            TYPE    ,$CRLF          ;YES-TYPE CRLF
7895  033560  105037  033740            CLRB    BOTFLG          ;CLEAR BOT FLAG
7896  033564  105037  033741            CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7897  033570  013702  033736            MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
7898  033574  010237  033544            MOV     R2,11$          ;SETUP 11$ FOR TYPING
7899  033600  112742  000040            MOVB    #' ,-(R2)       ;RESTORE SPACE
7900  033604  105722                    TSTB    (R2)+           ;RESTORE R2
7901  033606  000677                    BR      5$              ;TYPE REST OF STRING
7902  033610                    12$:
7903                          ;TYPE ERROR HEADER AND ERROR DATA
```

```
7904  033610                          13$:
7905  033610  016001  000002                  MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
7906  033614  001444                          BEQ     21$             ;BRANCH IF NO HEADER
7907  033616  104401  001217                  TYPE    $CRLF           ;(ASSUME NO DATA)
7908  033622  016002  000004                  MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
7909  033626  016003  000006                  MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
7910  033632  012137  033642          14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
7911  033636  001433                          BEQ     21$             ;BRANCH IF END OF HEADERS
7912                                                                  ;(ASSUME END OF DATA)
7913  033640  104401                          TYPE
7914  033642  000000                  15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
7915  033644  104401  001217                  TYPE    $CRLF
7916  033650  005702                          TST     R2              ;DATA WITH HEADER??
7917  033652  001767                          BEQ     14$             ;NO!!
7918  033654  012204                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
7919  033656  012205                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
7920  033660  105725                  16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
7921  033662  100407                          BMI     18$             ;BINARY
7922  033664  001403                          BEQ     17$             ;OCTAL
7923  033666  013446                          MOV     @(R4)+,-(SP)    ;DECIMAL
7924  033670  104405                          TYPDS
7925  033672  000405                          BR      19$
7926  033674  013446                  17$:    MOV     @(R4)+,-(SP)
7927  033676  104402                          TYPOC
7928  033700  000402                          BR      19$
7929  033702  013446                  18$:    MOV     @(R4)+,-(SP)
7930  033704  104406                          TYPBN
7931  033706  005714                  19$:    TST     (R4)            ;MORE DATA??
7932  033710  001403                          BEQ     20$             ;NO!!
7933  033712  104401  033775                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
7934  033716  000760                          BR      16$             ;AND CONTINUE
7935  033720  104401  001217          20$:    TYPE    $CRLF           ;TYPE ONE BLANK LINE
7936  033724  000742                          BR      14$             ;BEFORE NEXT HEADER
7937  033726  104415                  21$:    RESREG
7938  033730  000207                          RTS     PC
7939
7940  033732  000000                  TSTNMB: .WORD   0               ;TEST NUMBER
7941  033734  000000                  ERRNMB: .WORD   0               ;ERROR NUMBER
7942  033736  000000                  BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
7943  033740     000                  BOTFLG: .BYTE   0               ;BOT FLAG
7944  033741     000                  CHRCNT: .BYTE   0               ;CHARACTER COUNT
7945
7946  033742  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
7947  033750  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
7948  033756  000043
7949  033760  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
7950  033766     000
7951  033767     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
7952  033774     000
7953  033775     040  000040          ERTY04: .ASCIZ  @ @
7954                                  .EVEN
7955
```

```
                                .SBTTL  TEST PREPARATION MODULE
                                ;
                                ; THIS MODULE PREPARES THE RMO3 SUBSYSTEM FOR THE EXECUTION OF A TEST,
                                ; REPORTING AN ERROR TO THE USER IF AN ERROR IS  DETECTED.  THE USER
                                ; SPECIFIES TASKS TO BE PERFORMED, WHICH THE MODULE EXECUTES
                                ; USING SUBROUTINES.
                                ;
                                ;CALL:
                                ;       JSR     PC,TSTPRP
                                ;       .WORD                   TASK/VERIFY DESCRIPTOR
                                ;       BR      ??              RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE IF ERROR
                                ;       ERROR                   ERROR DEFINED BY MODULE
                                ;
                                ;TASK/VERIFY DESCRIPTOR
                                ;       BIT 15 = 1      SELECT DEVICE AND VERIFY DEVICE IS AVAILABLE
                                ;       BIT 14 = 1      CLEAR CONTROLLER AND SELECT DEVICE
                                ;       BIT 13          (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 12 = 1      PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;       BIT 11 = 1      RECALIBRATE IF POSITIONING IN PROGRESS
                                ;       BIT 10
                                ;       BIT 9
                                ;       BIT 8
                                ;       BIT 7
                                ;       BIT 6  = 1      VERIFY CONTROLLER CLEAR OPERATION
                                ;       BIT 5           (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 4  = 1      VERIFY PACK ACKNOWLEDGE
                                ;       BIT 3  = 1      VERIFY RECALIBRATION
                                ;       BIT 2
                                ;       BIT 1
                                ;       BIT 0
                                ;
034000                          TSTPRP:
                                ;
                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
034000  017637  000000  034712          MOV     @(SP),500$      ;STORE DESCRIPTOR
034006  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
034012  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
034016  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
034022  032737  100000  034712          BIT     #BIT15,500$     ;SELECT DEVICE??
034030  001414                          BEQ     30$             ;NO!!
034032  004737  045206                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
034036  000411                          BR      30$             ;NO ERROR - CONTINUE
034040  000401                          BR      20$
034042  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
034044  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
034050  113776  034042  000000          MOVB    10$,@(SP)
034056  000137  034700                  JMP     400$
034062                          30$:
                                ;
                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
034062  032737  040000  034712          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
034070  001453                          BEQ     120$            ;NO!!
```

```
8012  034072  004737  046660            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
8013  034076  000411                    BR      60$             ;CONTINUE - NO ERROR
8014  034100  000401                    BR      50$
8015  034102  000000            40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
8016  034104  062716  000004    50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8017  034110  113776  034102  000000    MOVB    40$,@(SP)
8018  034116  000137  034700            JMP     400$
8019  034122                    60$:
8020                                    ;;*****************************************************************
8021                                    ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
8022  034122  032737  000100  034712    BIT     #BIT6,500$      ;VERIFY??
8023  034130  001433                    BEQ     120$            ;NO!!
8024  034132  004737  037432            JSR     PC,GETSTS       ;SETUP INDEX TABLE
8025  034136  004737  037516            JSR     PC,GET          ;GO GET STATUS
8026  034142  000411                    BR      90$             ;NO ERROR GETTING STATUS
8027  034144  000401                    BR      80$
8028  034146  000000            70$:    .WORD                   ;ERROR FROM GETTING STATUS
8029  034150  062716  000004    80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8030  034154  113776  034146  000000    MOVB    70$,@(SP)
8031  034162  000137  034700            JMP     400$
8032  034166  004737  046776    90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
8033  034172  000412                    BR      120$            ;NO ERROR IN CLEAR
8034  034174  000401                    BR      110$
8035  034176  000000            100$:   .WORD                   ;ERROR IN STATUS CLEAR
8036  034200  005726            110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
8037  034202  062716  000004            ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
8038  034206  113776  034176  000000    MOVB    100$,@(SP)      ;ERROR TO USER
8039  034214  000137  034700            JMP     400$
8040  034220                    120$:
8041
8042                                    ;;*****************************************************************
8043                                    ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
8044                                    ;NOT VALID
8045  034220  032737  010000  034712    BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
8046  034226  001511                    BEQ     240$            ;NO!!
8047  034230  112737  000012  001506    MOVB    #RMDS,GETINX    ;GET RMDS
8048  034236  112737  000200  001507    MOVB    #200,GETINX+1
8049  034244  004737  037516            JSR     PC,GET
8050  034250  000411                    BR      150$            ;NO ERROR GETTING RMDS
8051  034252  000401                    BR      140$
8052  034254  000000            130$:   .WORD
8053  034256  062716  000004    140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8054  034262  113776  034254  000000    MOVB    130$,@(SP)
8055  034270  000137  034700            JMP     400$
8056  034274  032737  000100  001342  150$:   BIT     #VV,RMDSI       ;IS VOLUME VALID??
8057  034302  001063                    BNE     240$            ;YES!!
8058  034304  005037  001474            CLR     MEDENB          ;CLEAR MEDIA ENABLE
8059  034310  012737  000023  001400    MOV     #PAKACK!GO,RMCS10       ;LOAD PACK ACK COMMAND
8060  034316  112737  000000  001535    MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
8061  034324  112737  000200  001536    MOVB    #200,PUTINX+1
8062  034332  004737  037766            JSR     PC,PUT          ;GO WRITE COMMAND
8063  034336  000410                    BR      180$            ;NO ERROR LOADING REGISTER
8064  034340  000401                    BR      170$
8065  034342  000000            160$:   .WORD                   ;ERROR FROM PUT SUB
8066  034344  062716  000004    170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8067  034350  113776  034342  000000    MOVB    160$,@(SP)
```

```
 8068  034356  000550                        BR      400$
 8069  034360                        180$:
 8070
 8071                                ;;*****************************************************************
 8072                                ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
 8073  034360  032737  000020  034712        BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
 8074  034366  001431                        BEQ     240$            ;NO!!
 8075  034370  004737  037432                JSR     PC,GETSTS       ;SETUP FOR STATUS
 8076  034374  004737  037516                JSR     PC,GET          ;GO GET STATUS
 8077  034400  000410                        BR      210$            ;NO ERROR GETTING STATUS
 8078  034402  000401                        BR      200$
 8079  034404  000000                190$:   .WORD                   ;ERROR FROM GET SUB
 8080  034406  062716  000004        200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8081  034412  113776  034404  000000        MOVB    190$,@(SP)
 8082  034420  000527                        BR      400$
 8083  034422  004737  047656        210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
 8084  034426  000411                        BR      240$            ;NO ERROR
 8085  034430  000401                        BR      230$
 8086  034432  000000                220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
 8087  034434  005726                230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
 8088  034436  062716  000004                ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8089  034442  113776  034432  000000        MOVB    220$,@(SP)
 8090  034450  000513                        BR      400$
 8091  034452                        240$:
 8092
 8093                                ;;*****************************************************************
 8094                                ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
 8095  034452  032737  004000  034712        BIT     #BIT11,500$     ;RECALIBRATE??
 8096  034460  001513                        BEQ     410$            ;NO!!
 8097  034462  112737  000012  001506        MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
 8098  034470  112737  000200  001507        MOVB    #200,GETINX+1
 8099  034476  004737  037516                JSR     PC,GET          ;GO GET RMDS
 8100  034502  000410                        BR      270$            ;NO ERROR GETTING RMDS
 8101  034504  000401                        BR      260$
 8102  034506  000000                250$:   .WORD                   ;ERROR FROM GET SUB
 8103  034510  062716  000004        260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8104  034514  113776  034506  000000        MOVB    250$,@(SP)
 8105  034522  000466                        BR      400$
 8106  034524  032737  020000  001342  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
 8107  034532  001466                        BEQ     410$            ;NO!!
 8108  034534  012737  000007  001400        MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
 8109  034542  112737  000000  001535        MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
 8110  034550  112737  000200  001536        MOVB    #200,PUTINX+1
 8111  034556  004737  037766                JSR     PC,PUT          ;GO ISSUE RECALIBRATE
 8112  034562  000410                        BR      300$            ;NO ERROR
 8113  034564  000401                        BR      290$
 8114  034566  000000                280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
 8115  034570  062716  000004        290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8116  034574  113776  034566  000000        MOVB    280$,@(SP)
 8117  034602  000436                        BR      400$
 8118  034604  004737  037432        300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
 8119  034610  004737  040326                JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
 8120
 8121                                ;;*****************************************************************
 8122                                ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
 8123  034614  032737  000010  034712        BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 8124  034622  001432                        BEQ     410$            ;NO!!
 8125  034624  004737  037516                JSR     PC,GET          ;GO GET STATUS
 8126  034630  000410                        BR      330$            ;NO ERROR GETTING STATUS
 8127  034632  000401                        BR      320$
 8128  034634  000000              310$:     .WORD                   ;ERROR FROM GET
 8129  034636  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8130  034642  113776  034634  000000        MOVB    310$,@(SP)
 8131  034650  000413                        BR      400$
 8132  034652  004737  050452      330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 8133  034656  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
 8134  034660  000401                        BR      350$
 8135  034662  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
 8136  034664  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
 8137  034666  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8138  034672  113776  034662  000000        MOVB    340$,@(SP)
 8139  034700  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 8140  034704  000240                        NOP
 8141  034706  000240                        NOP
 8142  034710  000207              410$:     RTS     PC              ;RETURN TO USER
 8143
 8144  034712  000000              500$:     .WORD                   ;TASK/VERIFY DESCRIPTOR
```

```
8145                                    .SBTTL  BAD SECTOR MODULE
8146
8147                                    ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS, I.E.,
8148                                    ;       RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK, AND,
8149                                    ;       APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
8150                                    ;THE BAD SECTOR FILE OR ASSIGN A NEW SECTOR IF THE ONE ELECTED IS
8151                                    ;NOT AVAILABLE FOR USE.
8152                                    ;
8153                                    ;INFORMATION REQUIRED BY THE MODULE INCLUDES
8154                                    ;       .RMDCO, THE DESIRED CYLINDER,
8155                                    ;       .RMDAO, THE TRACK AND SECTOR ADDRESS,
8156                                    ;       .RMWCO, THE WORD COUNT,
8157                                    ;       .RMCS10, THE COMMAND.
8158                                    ;
8159                                    ;MODULE CALL IS AS FOLLOWS,
8160                                    ;
8161                                    ;       JSR     PC,BADSCT
8162                                    ;       BR      ???             ;RETURN HERE IF NO ERROR
8163                                    ;       TYPE    ,MESSAGE        ;RETURN HERE IF THE BAD SECTOR FILE
8164                                    ;                               CANNOT BE RECOVERED
8165                                    ;       ERROR                   THE MODULE DEFINES THE ERROR NUMBER
8166                                    ;
8167                                    ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
8168                                    ;GENERATOR SUBROUTINE, AND PRESERVES THE "PUT BUFFER" SO THAT THE
8169                                    ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
8170                                    ;OPERATION.
8171                                    ;
8172                                    ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
8173                                    ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS "ASNDA" AND "ASNDC"
8174                                    ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.
8175
8176
8177  034714                            BADSCT:
8178
8179                                    ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
8180                                    ;HAVE BEEN RECOVERED.
8181  034714  005737  001474                  TST     MEDENB
8182  034720  001402                        BEQ     10$
8183  034722  000137  036140                JMP     300$            ;BAD SECTOR FILE IS AVAILABLE
8184
8185                                    ;;*****************************************************************
8186                                    ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
8187  034726                            10$:
8188
8189                                    ;SAVE THE USER'S PUT BUFFER
8190  034726  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8191  034730  005000                        CLR     R0
8192  034732  016060  001400  103566  20$:    MOV     PUTBUF(R0),BUFFER(R0)
8193  034740  062700  000002                ADD     #2,R0           ;ADVANCE TO NEXT BUFFER POSITION
8194  034744  022700  000046                CMP     #46,R0          ;END OF BUFFER
8195  034750  103370                        BHIS    20$             ;NO !!
8196
8197                                    ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
8198  034752  012737  000003  036605        MOV     #3,500$         ;RETRY COUNT
8199  034760  012737  002000  001406        MOV     #002000,RMDAO   ;STARTING SECTOR ADDRESS
8200  034766  012737  001466  001434        MOV     #822.,RMDCO     ;DESIRED CYLINDER
```

```
8201  035774  012737  177376  001402          MOV     #<↑C258.+1>,RMWCO       ;WORD COUNT
8202  035002  012737  010000  001432          MOV     #FMT16,RMOFO    ;16 BIT FORMAT
8203  035010  012737  101546  001404          MOV     #MFGFIL,RMBAO   ;BUFFER ADDRESS
8204
8205  035016  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
8206  035022  112720  000006                  MOVB    #RMDA,(R0)+
8207  035026  112720  000034                  MOVB    #RMDC,(R0)+
8208  035032  112720  000032                  MOVB    #RMWC,(R0)+
8209  035036  112720  000032                  MOVB    #RMOF,(R0)+
8210  035042  112720  000004                  MOVB    #RMBA,(R0)+
8211  035046  112720  000000                  MOVB    #RMCS1,(R0)+
8212  035052  112720  000200                  MOVB    #200,(R0)+
8213  035056  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8214
8215                          ;SET GET INDEX TABLE FOR READING STATUS
8216  035060  004737  037432                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
8217  035064                  30$:
8218
8219                          ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
8220  035064  012737  000011  001400          MOV     #DRVCLR!GO,RMCS1O       ;LOAD COMMAND IN PUT BUFFER
8221  035072  004737  037766                  JSR     PC,PUT          ;OUTPUT COMMAND
8222  035076  000411                          BR      45$             ;RETURN HERE IF NO ERROR
8223  035100  000401                          BR      40$             ;GET AROUND ERROR #
8224  035102  000000                  35$:    .WORD                   ;ERROR # GOES HERE
8225
8226  035104  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8227  035110  113776  035102  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
8228  035116  000137  035616                  JMP     215$
8229  035122  004737  037516          45$:    JSR     PC,GET          ;GO GET STATUS
8230  035126  000411                          BR      60$             ;RETURN HERE IF NO ERROR
8231  035130  000401                          BR      55$             ;GET AROUND ERROR #
8232  035132  000000                  50$:    .WORD                   ;ERROR # GOES HERE
8233
8234  035134  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8235  035140  113776  035132  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
8236  035146  000137  035616                  JMP     215$
8237
8238  035152  004737  052214          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
8239  035156  000412                          BR      75$             ;RETURN HERE IF NO ERROR
8240  035160  000401                          BR      70$             ;GET AROUND ERROR
8241  035162  000000                  65$:    .WORD                   ;ERROR # GOES HERE
8242
8243  035164  005726                  70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
8244  035166  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8245  035172  113776  035162  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
8246  035200  000137  035616                  JMP     215$
8247
8248  035204                  75$:
8249
8250                          ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
8251  035204  032737  000100  001342          BIT     #VV,RMDSI       ;IS VV RESET ??
8252  035212  001050                          BNE     120$            ;NO !!
8253  035214  012737  000023  001400          MOV     #PACACK!GO,RMCS1O       ;LOAD COMMAND
8254  035222  004737  037766                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
8255  035226  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
8256  035230  000401                          BR      85$             ;GET AROUND ERROR #
```

```
8257  035232  000000                  80$:    .WORD                   ;ERROR # GOES HERE
8258
8259  035234  062716  000006          85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8260  035240  113776  035232  000000          MOVB    80$,@(SP)       ;MOVE ERROR # TO ERROR CALL
8261  035246  000137  035616                  JMP     215$
8262  035252  004737  037516          90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
8263  035256  000411                          BR      105$            ;RETURN HERE IF NO ERROR
8264  035260  000401                          BR      100$            ;GET AROUND ERROR #
8265  035262  000000          95$:    .WORD                           ;ERROR # GOES HERE
8266
8267  035264  062716  000006          100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8268  035270  113776  035262  000000          MOVB    95$,@(SP)       ;MOVE ERROR # TO CALL
8269  035276  000137  035616                  JMP     215$
8270
8271  035302  004737  047656          105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
8272  035306  000412                          BR      120$            ;RETURN HERE IF NO ERROR
8273  035310  000401                          BR      115$            ;GET AROUND ERROR #
8274  035312  000000          110$:   .WORD                           ;ERROR # GOES HERE
8275
8276  035314  005726          115$:   TST     (SP)+                   ;STRIP RETURN TO SUBROUTINE
8277  035316  062716  000006          ADD     #6,(SP)                 ;MOVE SP TO USERS ERROR CALL
8278  035322  113776  035312  000000  MOVB    110$,@(SP)              ;MOVE ERROR # TO USERS ERROR CALL
8279  035330  000137  035616          JMP     215$
8280
8281  035334                  120$:
8282
8283                          ;RECALIBRATE THE DRIVE IF PIP IS SET
8284  035334  032737  020000  001342  BIT     #PIP,RMDSI              ;IS PIP SET ??
8285  035342  001452                  BEQ     165$                    ;NO !!
8286
8287  035344  012737  000007  001400  MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
8288  035352  004737  037766          JSR     PC,PUT                  ;PUT THE RECAL COMMAND
8289  035356  000411                  BR      135$                    ;RETURN HERE IF NO ERROR
8290  035360  000401                  BR      130$                    ;GET AROUND ERROR #
8291  035362  000000          125$:   .WORD                           ;ERROR # GOES HERE
8292
8293  035364  062716  000006          130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8294  035370  113776  035362  000000          MOVB    125$,@(SP)      ;MOVE ERROR # TO USERS CALL
8295  035376  000137  035616                  JMP     215$
8296  035402                          135$:
8297  035402  004737  040326                  JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
8298  035406  004737  037516                  JSR     PC,GET          ;GO GET RECAL STATUS
8299  035412  000411                          BR      150$            ;RETURN HERE IF NO ERROR
8300  035414  000401                          BR      145$            ;GET AROUND ERROR #
8301  035416  000000          140$:   .WORD                           ;ERROR # GOES HERE
8302
8303  035420  062716  000006          145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8304  035424  113776  035416  000000          MOVB    140$,@(SP)      ;MOVE ERROR TO USERS CALL
8305  035432  000137  035616                  JMP     215$
8306
8307  035436  004737  050452          150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
8308  035442  000412                          BR      160$            ;RETURN HERE IF NO ERROR
8309  035444  000401                          BR      155$            ;GET AROUND ERROR #
8310  035446  000000          155$:   .WORD                           ;ERROR # GOES HERE
8311
8312  035450  005726          160$:   TST     (SP)+                   ;STRIP RETURN TO SUBROUTINE
```

```
8313  035452  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8314  035456  113776  035446  000000    MOVB    155$,2(SP)       ;MOVE ERROR # TO USERS CALL
8315  035464  000137  035616            JMP     215$
8316
8317  035470                    165$:
8318
8319                              ;READ THE SECTOR IDENTIFIED BY RMDA0, INCLUDING HEADER AND DATA
8320
8321  035470  012737  000073  001400    MOV     #RH!GO,RMCS10    ;LOAD READ HEADER AND DATA COMMAND
8322  035476  004737  037766            JSR     PC,PUT           ;PUT COMMAND
8323  035502  000411                    BR      180$             ;RETURN HERE IF NO ERROR
8324  035504  000401                    BR      175$             ;GET AROUND ERROR #
8325  035506  000000            170$:   .WORD                    ;ERROR # GOES HERE
8326
8327  035510  062716  000006    175$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8328  035514  113776  035506  000000    MOVB    170$,2(SP)       ;MOVE ERROR # TO USERS ERROR CALL
8329  035522  000137  035616            JMP     215$
8330
8331  035526  004737  040326    180$:   JSR     PC,TIMOUT        ;WAIT FOR READ OPERATION TO COMPLETE
8332  035532  004737  037516            JSR     PC,GET           ;GO GET STATUS FOR READ OPERATION
8333  035536  000411                    BR      195$             ;RETURN HERE IF NO ERROR
8334  035540  000401                    BR      190$             ;GET AROUND ERROR #
8335  035542  000000            185$:   .WORD                    ;ERROR # GOES HERE
8336
8337  035544  062716  000006    190$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8338  035550  113776  035542  000000    MOVB    185$,2(SP)       ;MOVE ERROR # TO CALL
8339  035556  000137  035616            JMP     215$
8340
8341  035562  004737  053016    195$:   JSR     PC,DTASTS        ;GO VERIFY RESULTS OF READ OPERATION
8342  035566  000412                    BR      210$             ;RETURN HERE IF NO ERROR
8343  035570  000401                    BR      205$             ;GET AROUND ERROR #
8344  035572  000000            200$:   .WORD                    ;ERROR # GOES HERE
8345
8346  035574  005726    205$:   TST     (SP)+            ;STRIP RETURN ADDRESS TO SUBROUTINE
8347  035576  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8348  035602  113776  035572  000000    MOVB    200$,2(SP)       ;MOVE ERROR # TO USERS CALL
8349  035610  000137  035616            JMP     215$
8350
8351  035614  000456            210$:   BR      240$
8352
8353  035616                    215$:
8354
8355                              ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
8356                              ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
8357
8358  035616  032737  010000  001342    BIT     #MOL,RMDSI       ;IS MEDIUM ON LINE ??
8359  035624  001446                    BEQ     230$             ;YES !!
8360
8361  035626  005337  036606            DEC     500$             ;DECREMENT RETRY COUNT
8362  035632  100037                    BPL     225$             ;RETRY IF COUNT NOT NEG
8363
8364                              ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
8365  035634  013746  001344            MOV     RMER1I,-(SP)     ;GET ER1
8366  035640  042716  100720            BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
8367  035644  032737  000010  001372    BIT     #DPE,RMER2I      ;WAS THER A DATA PARITY ERROR ??
8368  035652  001402                    BEQ     220$             ;NO !!
```

```
8369  035654  042716  000010            BIC     #PAR,(SP)          ;YES - CLEAR PAR
8370  035660  005726                220$: TST     (SP)+              ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
8371  035662  001027                      BNE     230$               ;YES !!
8372  035664  013746  001372              MOV     RMER2I,-(SP)       ;GET ER2
8373  035670  042726  100010              BIC     #BSE!DPE,(SP)+     ;CLEAR MEDIA ERRORS
8374  035674  001022                      BNE     230$               ;BRANCH IF NONMEDIA ERRORS DETECTED
8375
8376                                ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
8377                                ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
8378                                ;ANOTHER AREA ON THE LAST TRACK
8379
8380  035676  062737  000002  001406      ADD     #2,RMDAO           ;ADVANCE SECTOR ADDRESS
8381  035704  122737  000012  001406      CMPB    #10.,RMDAO         ;QUIT IF ALL MFG SECTORS HAVE BEEN
8382  035712  001413                      BEQ     230$               ;TRIED
8383  035714  122737  000040  001406      CMPB    #32.,RMDAO         ;QUIT IF ALL USER SECTORS HAVE
8384  035722  001407                      BEQ     230$               ;BEEN TRIED
8385  035724  012737  000003  036606      MOV     #3,500$            ;REINSTATE RETRY COUNT FOR THIS SECTOR
8386  035732  162716  000006        225$: SUB     #6,(SP)            ;MOVE SP BACK TO NO ERROR
8387  035736  000137  035064              JMP     30$                ;RETRY THE READ OPERATION
8388
8389  035742                        230$:
8390
8391                                ;THE BAD SECTOR FILE CANNOT BE READ
8392  035742  162716  000004              SUB     #4,(SP)            ;MOVE SP TO ERROR RETURN
8393  035746  000137  036602              JMP     410$               ;GO TO MODULE EXIT
8394
8395  035752                        240$:
8396
8397                                ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
8398                                ;       READ THE USER FILE IF THIS IS THE MFG FILE
8399                                ;       OR ELSE DONE
8400
8401  035752  122737  000011  001406      CMPB    #9.,RMDAO          ;WAS THE USER FILE READ ??
8402  035760  103451                      BLO     260$               ;YES - READ IS COMPLETE
8403
8404                                ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
8405                                ;       ELSE DUMMY THE BAD SECTOR FILE
8406  035762  032737  140000  101546      BIT     #MSE!USE,MFGFIL    ;ARE THE BAD SECTOR FLAGS OFF ??
8407  035770  001410                      BEQ     250$               ;YES - 144 IS DISABLED
8408
8409  035772  012737  002012  001406      MOV     #002012,RMDAO      ;READ THE USER FILE
8410  036000  012737  000003  036606      MOV     #3,500$            ;RELOAD THE RETRY COUNT FOR THIS SECTOR
8411  036006  000137  035064              JMP     30$                ;GO READ THE USER FILE
8412  036012                        250$:
8413
8414                                ;DUMMY THE BAD SECTOR FILES
8415  036012  010046                      MOV     R0,-(SP)           ;;PUSH R0 ON STACK
8416  036014  010146                      MOV     R1,-(SP)           ;;PUSH R1 ON STACK
8417  036016  012701  000374              MOV     #252.,R1           ;R1 = NUMBER OF ENTRIES IN FILES
8418  036022  012700  000014              MOV     #14,R0             ;R0 = ACCESS INDEX TO FILE  STORAGE
8419  036026  012760  177777  101546 255$: MOV     #-1,MFGFIL(R0)     ;ENTER ALL ONES IN MFG FILE
8420  036034  012760  177777  102556      MOV     #-1,USRFIL(R0)     ;ENTER ALL ONES IN USER FILE
8421  036042  062700  000002              ADD     #2,R0              ;ADVANCE ADDRESS
8422  036046  005301                      DEC     R1                 ;DECREMENT COUNT
8423  036050  001366                      BNE     255$               ;CONTINUE IF NOT DONE
8424  036052  012701  000006              MOV     #6.,R1             ;CLEAR ID, SERIAL NUMBER ETC
```

```
8425  036056  005000                  256$:   CLR     R0
8426  036060  005060  101546                  CLR     MFGFIL(R0)
8427  036064  005060  102556                  CLR     USRFIL(R0)
8428  036070  062700  000002                  ADD     #2,R0
8429  036074  005301                          DEC     R1
8430  036076  001370                          BNE     256$
8431
8432  036100  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8433  036102  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8434  036104                          260$:
8435
8436                                  ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
8437  036104  012737  177777  001474          MOV     #-1,MEDENB
8438
8439  036112  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8440  036114  005000                          CLR     R0              ;R0 IS REGISTER INDEX
8441  036116  016060  103566  001400  265$:   MOV     BUFFER(R0),PUTBUF(R0)
8442  036124  062700  000002                  ADD     #2,R0           ;ADVANCE R0
8443  036130  022700  000046                  CMP     #46,R0          ;DONE ??
8444  036134  103370                          BHIS    265$
8445
8446  036136  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8447  036140                          270$:
8448
8449  036140                          300$:
8450
8451                                  ;;*****************************************************************
8452                                  ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
8453                                  ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
8454                                  ;SECTOR IS IN EITHER FILE.
8455
8456                                  ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
8457  036140  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8458  036142  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8459  036144  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8460  036146  013737  001434  001476          MOV     RMDCO,CYLNDC    ;LOAD REQUESTED CYLINDER, TRACK,
8461  036154  013737  001406  001500          MOV     RMDAD,SCTNDA    ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
8462  036162  005002                          CLR     R2              ;R2 = NUMBER OF SECTORS
8463  036164  013700  001402                  MOV     RMWCO,R0        ;R0 = WORD COUNT
8464  036170  005300                          DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
8465  036172  005100                          COM     R0
8466  036174  012701  000400                  MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
8467  036200  032737  000002  001400          BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
8468  036206  001402                          BEQ     305$            ;NO !!
8469  036210  012701  000402                  MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
8470  036214  020100                  305$:   CMP     R1,R0           ;IS THERE A FULL SECTOR ??
8471  036216  101404                          BLOS    310$            ;YES !!
8472  036220  005700                          TST     R0              ;IS R0 ZERO ??
8473  036222  001405                          BEQ     315$            ;YES !!
8474  036224  005202                          INC     R2              ;INCREMENT FOR PARTIAL SECTOR
8475  036226  000403                          BR      315$
8476  036230  160100                  310$:   SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
8477  036232  005202                          INC     R2              ;INCREMENT SECTOR COUNT
8478  036234  000767                          BR      305$
8479  036236  010237  036606          315$:   MOV     R2,SOO$         ;SAVE SECTOR COUNT
8480
```

```
8481  036242                                  316$:
8482
8483                                          ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
8484                                          ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
8485                                          ;MORE THAN ONE.
8486
8487  036242  012737  101562  036616          MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
8488                                                                  ;FILE TO BASE ADDRESS STORAGE
8489  036250  013737  001476  036612  320$:   MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
8490  036256  013737  001500  036614          MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
8491  036264  013737  036606  036610          MOV     510$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
8492
8493                                          ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
8494                                          ;CYLINDER, TRACK AND SECTOR ADDRESS
8495  036272  013700  036616          325$:   MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
8496  036276  012701  000376                  MOV     #((127.*4)/2),R1        ;R1 = LENGTH OF FILE
8497  036302  060100                          ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
8498  036304  021037  036612          330$:   CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
8499  036310  001405                          BEQ     345$            ;YES !!
8500  036312  101002                          BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
8501  036314                          335$:
8502
8503                                          ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
8504                                          ;SO ADD DISPLACEMENT TO CURRENT POSITION
8505  036314  060100                          ADD     R1,R0
8506  036316  000410                          BR      350$
8507  036320                          340$:
8508
8509                                          ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
8510                                          ;FROM CURRENT POSITION
8511  036320  160100                          SUB     R1,R0
8512  036322  000406                          BR      350$
8513  036324                          345$:
8514
8515                                          ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
8516                                          ;THE COMPARING TRACK, AND SECTOR.
8517  036324  026037  000002  036614          CMP     2(R0),530$      ;ARE THEY EQUAL ??
8518  036332  001413                          BEQ     350$            ;YES !!
8519  036334  101371                          BHI     340$            ;BRANCH IF HIGHER
8520  036336  000766                          BR      335$            ;BRANCH IF LOWER
8521  036340                          350$:
8522
8523                                          ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
8524                                          ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
8525  036340  005701                          TST     R1              ;IS DISPLACEMENT ZERO ??
8526  036342  001440                          BEQ     370$            ;YES !!
8527  036344  006201                          ASR     R1              ;HALVE THE DISLPACEMENT
8528  036346  042701  000003                  BIC     #3,R1
8529  036352  020037  036616                  CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
8530  036356  103432                          BLO     370$            ;NO !!
8531  036360  000751                          BR      330$            ;CONTINUE SEARCH
8532  036362                          360$:
8533
8534                                          ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
8535                                          ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
8536  036362  105237  001500                  INCB    ASNDA           ;INCREMENT SECTOR
```

```
8537  036456  122737  000037  001500          CMPB    #31.,ASNDA      ;SECTOR OK ??
8538  036464  103022                          BHIS    365$            ;YES !!
8539  036466  105037  001500                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
8540  036472  105237  001501                  INCB    ASNDA+1
8541  036476  122737  000004  001501          CMPB    #4,ASNDA+1      ;TRACK OK ??
8542  036504  103012                          BHIS    365$            ;YES !!
8543  036506  005037  001500                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
8544  036512  005237  001476                  INC     ASNDC           ;INCREMENT CYLINDER
8545  036516  022737  001466  001476          CMP     #822.,ASNDC     ;CYLINDER OK ??
8546  036524  103002                          BHIS    365$            ;YES !!
8547  036526  005037  001476                  CLR     ASNDC
8548  036532  000677                  365$:   BR      316$            ;REPEAT SEARCH
8549
8550  036534                          370$:
8551
8552                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
8553                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
8554                                  ;IS NOT ZERO.
8555
8556  036534  005337  036610                  DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
8557  036540  001432                          BEQ     380$            ;DONE IF ZERO
8558  036542  105237  036614                  INCB    530$            ;INCREMENT THE COMPARING SECTOR
8559  036546  122737  000037  036614          CMPB    #31.,530$       ;SECTOR OK ??
8560  036554  103022                          BHIS    375$            ;YES !!
8561  036556  105037  036614                  CLRB    530$            ;CLEAR SECTOR
8562  036562  105237  036615                  INCB    530$+1          ;INCREMENT TRACK
8563  036566  122737  000004  036615          CMPB    #4,530$+1       ;TRACK OK ??
8564  036574  103012                          BHIS    375$            ;YES !!
8565  036576  005037  036614                  CLR     530$            ;CLEAR SECTOR TRACK
8566  036602  005237  036612                  INC     520$            ;INCREMENT CYLINDER
8567  036606  022737  001466  036612          CMP     #822.,520$      ;CYLINDER OK ??
8568  036614  103002                          BHIS    375$            ;YES !!
8569  036616  005037  036612                  CLR     520$            ;CLEAR CYLINDER
8570  036622  000137  036272          375$:   JMP     325$            ;SEARCH NEXT SECTOR
8571
8572  036626                          380$:
8573
8574                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
8575                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
8576                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
8577
8578  036626  022737  102572  036616          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
8579  036634  001405                          BEQ     400$            ;DONE IF BASE = USER
8580  036636  012737  102572  036616          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
8581  036644  000137  036250          390$:   JMP     320$            ;SEARCH USER FILE
8582
8583
8584  036650                          400$:
8585
8586                                  ;ASSIGN THE SECTOR AND RETURN TO USER
8587  036650  013737  001476  001434          MOV     ASNDC,RMDCO     ;LOAD CYLINDER
8588  036656  013737  001500  001406          MOV     ASNDA,RMDAO     ;LOAD TRACK AND SECTOR
8589  036664  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8590  036666  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8591  036670  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8592  036672  000240                  410$:   NOP
```

```
 8593  036604  000207                    RTS     PC
 8594
 8595                              ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
 8596
 8597  036606  000000              500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 8598  036610  000000              510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
 8599  036612  000000              520$:   .WORD   ;COMPARING CYLINDER
 8600  036614  000000              530$:   .WORD   ;COMPARING TRACK AND SECTOR
 8601  036616  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

036620                          GENBUF:
036620  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036622  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036624  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036626  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
036630  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
036632  013700  001404                  MOV     RMBA0,R0        ;LOAD DATA BUFFER ADDRESS
036636  013701  001402                  MOV     RMWC0,R1        ;LOAD WORD COUNT
036642  013737  001434  037060          MOV     RMDC0,60$       ;LOAD STARTING CYLINDER ADDRESS
036650  013737  001406  037062          MOV     RMDA0,65$       ;LOAD STARTING TRACK,SECTOR ADDRESS
036656  032737  000002  001400  10$:    BIT     #BIT1,RMCS10    ;WRITE HEADER & DATA??
036664  001450                          BEQ     25$             ;NO!!
036666  013710  037060                  MOV     60$,(R0)        ;WRITE HEADER WORD #1
036672  042710  176000                  BIC     #^CCYLMSK,(R0)
036676  052710  140000                  BIS     #MSE!USE,(R0)   ;SET (DISABLE) BAD SECTOR FLAGS
037002  012702  000035                  MOV     #29.,R2         ;R2 = MAXIMUM SECTOR ADDRESS
037006  032737  010000  001432          BIT     #FMT16,RMOF0    ;16 BIT FORMAT??
037014  001404                          BEQ     15$             ;NO!!
037016  052710  010000                  BIS     #FMT16,(R0)     ;SET FORMAT BIT IN HEADER
037022  012702  000037                  MOV     #31.,R2         ;CHANGE MAXIMUM SECTOR ADDRESS
037026  005201                  15$:    INC     R1              ;INCREMENT WORD COUNT
037030  001444                          BEQ     50$             ;EXIT IF DONE
037032  062700  000002                  ADD     #2,R0           ;MOVE R0 TO HEADER WORD #2
037036  013720  037062                  MOV     65$,(R0)+       ;WRITE HEADER WORD #2
037042  005201                          INC     R1              ;INCREMENT WORD COUNT AND
037044  001436                          BEQ     50$             ;EXIT IF DONE
037046  012703  037062                  MOV     #65$,R3         ;ADVANCE SECTOR ADDRESS
037052  105213                          INCB    (R3)
037054  120213                          CMPB    R2,(R3)         ;SECTOR OVERFLOW ??
037056  103013                          BHIS    25$             ;NO !!
037060  105013                          CLRB    (R3)            ;YES - CLEAR SECTOR ADDRESS
037062  105263  000001                  INCB    1(R3)           ;ADVANCE TRACK ADDRESS
037066  122763  000004  000001          CMPB    #4,1(R3)        ;TRACK OVERFLOW ??
037074  103004                          BHIS    25$             ;NO !!
037076  105063  000001                  CLRB    1(R3)           ;YES - CLEAR TRACK ADDRESS
037102  105237  037060                  INCB    60$             ;ADVANCE CYLINDER ADDRESS
037106  012704  000400          25$:    MOV     #256.,R4                ;LOAD SECTOR DATA COUNT
037112  013702  001174          30$:    MOV     $TMP0,R2        ;LOAD PATTERN ADDRESS
037116  013703  001176                  MOV     $TMP1,R3        ;LOAD PATTERN COUNT
037122  012220                  40$:    MOV     (R2)+,(R0)+     ;WRITE DATA PATTERN
037124  005201                          INC     R1              ;INCREMENT WORD COUNT AND
037026  001405                          BEQ     50$             ;EXIT IF DONE
037030  005304                          DEC     R4              ;DECREMENT SECTOR COUNT
```

```
8658  037032  001711                  BEQ     10$             ;START NEXT SECTOR IF 0
8659  037034  005303                  DEC     R3              ;DECREMENT PATTERN COUNT
8660  037036  001765                  BEQ     30$             ;RESTART PATTERN IF 0
8661  037040  000770                  BR      40$             ;CONTINUE DATA PATTERN
8662
8663  037042                  50$:
8664  037042  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
8665  037044  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
8666  037046  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
8667  037050  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8668  037052  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8669  037054  000240                  NOP
8670  037056  000207                  RTS     PC
8671
8672  037060  000000          60$:    .WORD                   ;CYLINDER ADDRESS STORAGE
8673  037062  000000          65$:    .WORD                   ;TRACK, SECTOR ADDRESS STORAGE
```

```
8674                                    .SBTTL  COMPARE BUFFER SUBROUTINE
8675
8676                                    ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
8677                                    ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
8678                                    ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
8679                                    ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
8680
8681                                    ;CALL:
8682                                    ;(1)    JSR     PC,CMPBUF
8683                                    ;(2)    .WORD   WRITE BUFFER ADDRESS
8684                                    ;       .WORD   READ BUFFER ADDRESS
8685                                    ;(3)    BR      ??              RETURN HERE IF NO ERROR
8686                                    ;(4)    NOP                     RETURN HERE IF ERROR
8687                                    ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
8688                                    ;(6)    ???
8689
8690  037064                            CMPBUF:
8691  037064  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8692  037066  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8693  037070  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8694  037072  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8695
8696  037074  005037  037430                   CLR     150$            ;CLEAR CORRECTION FLAG
8697                                    ;DETERMINE IF DATA SHOULD BE CORRECTED
8698  037100  032737  100000  001344           BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
8699  037106  001465                           BEQ     80$             ;NO !!
8700  037110  032737  000100  001344           BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
8701  037116  001061                           BNE     80$             ;YES !!
8702  037120  033737  004000  001362           BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
8703  037126  001055                           BNE     80$             ;NO !!
8704  037130  032737  010000  001362           BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
8705  037136  001451                           BEQ     80$             ;NO !!
8706
8707                                    ;CORRECT DATA USING ECC INFORMATION
8708  037140  013700  001404                   MOV     RMBAO,R0        ;R0 = STARTING BUFFER ADDRESS
8709  037144  013701  001374                   MOV     RMEC1I,R1       ;R1 = ECC POSITION
8710  037150  052737  100000  037430           BIS     #BIT15,150$     ;SET CORRECTION FLAG
8711                                    ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
8712
8713  037156  022701  000020            10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
8714  037162  103005                           BHIS    20$             ;NO !!
8715  037164  162701  000020                   SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
8716  037170  062700  000002                   ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
8717  037174  000770                           BR      10$
8718  037176  012702  000001            20$:    MOV     #1,R2           ;R2 = BIT POINTER
8719  037202  010203                           MOV     R2,R3           ;R3 = BIT NUMBER
8720                                    ;MOVE R2 TO STARTING BIT OF ERROR BURST
8721
8722  037204  020301            30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
8723  037206  001403                           BEQ     35$             ;YES !!
8724  037210  006302                           ASL     R2              ;SHIFT BIT POINTER
8725  037212  005203                           INC     R3              ;INCREMENT BIT NUMBER
8726  037214  000773                           BR      30$
8727  037216  012703  000013            35$:    MOV     #11.,R3         ;R3 = LENGTH OF ERROR BURST
8728
8729                                    ;CORRECT THE ERROR BURST
```

```
 8730  037222  030237  001376          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
 8731  037226  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
 8732  037230  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
 8733  037232  001402                          BEQ     50$             ;NO
 8734  037234  040210                          BIC     R2,(R0)         ;RESET THE BIT
 8735  037236  000401                          BR      60$
 8736  037240  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
 8737  037242  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
 8738  037244  001004                          BNE     70$
 8739  037246  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT OF NEXT WORD
 8740  037252  062700  000002                  ADD     #2,R0
 8741  037256  005303                  70$:    DEC     R3              ;END OF BURST ??
 8742  037260  001360                          BNE     40$             ;NO !!
 8743  037262  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
 8744  037266  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
 8745  037274  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
 8746  037300  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
 8747  037306  013702  001332                  MOV     RMWCI,R2                ;R2 = NUMBER OF WORDS TRANSFER
 8748  037312  163702  001402                  SUB     RMWCO,R2
 8749  037316  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
 8750  037320  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
 8751  037322  005302                          DEC     R2                      ;DECREMENT WORD COUNT
 8752  037324  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
 8753  037326  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
 8754  037330                          100$:
 8755  037330  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
 8756  037334  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
 8757  037340  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
 8758  037344  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
 8759  037350  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
 8760  037354  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
 8761  037362  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
 8762
 8763                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
 8764  037370  032737  100000  037430          BIT     #BIT15,150$             ;WAS ECC CORRECTION USED ??
 8765  037376  001403                          BEQ     105$                    ;NO !!
 8766  037400  112776  000163  000010          MOVB    #163,@10(SP)            ;ECC CORRECTION FAILED
 8767  037406  162766  000002  000010  105$:   SUB     #2,10(SP)                       ;MOVE SP TO RETURN IF ERROR
 8768  037414  000240                          NOP
 8769  037416                          110$:
 8770  037416  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
 8771  037420  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
 8772  037422  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
 8773  037424  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
 8774  037426  000207                          RTS     PC                              ;RETURN TO USER
 8775
 8776  037430  000000                  150$:   .WORD                           ;ECC CORRECTION FLAG
```

```
8777                                            .SBTTL  GET STATUS SUBROUTINE
8778                                            ;
8779                                            ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
8780                                            ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
8781                                            ;AND THEN RETURNS TO THE USER.
8782                                            ;
8783                                            ;CALL:  JSR     PC,GETSTS
8784                                            ;       ???                             RETURN HERE
8785
8786  037432                                    GETSTS:
8787  037432  010046                                    MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8788  037434  010146                                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8789  037436  010246                                    MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8790  037440  012700  001506                           MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
8791  037444  012701  001400                           MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
8792  037450  012702  000046                           MOV     #RMEC2,R2       ;R2 = REGISTER INDE
8793  037454  110220                    2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
8794  037456  005041                            CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
8795  037460  162702  000002            3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
8796  037464  100405                            BMI     4$              ;BRANCH OUT IF DONE
8797  037466  022702  000022                    CMP     #RMD8,R2        ;DONT WRITE RMD8 INDEX
8798  037472  001370                            BNE     2$
8799  037474  005041                            CLR     -(R1)
8800  037476  000770                            BR      3$
8801  037500  112720  000200            4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
8802  037504  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
8803  037506  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
8804  037510  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
8805  037512  000240                            NOP
8806  037514  000207                            RTS     PC
8807
```

```
                                .SBTTL  GET SUBROUTINE

                                ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
                                ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
                                ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
                                ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
                                ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
                                ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
                                ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
                                ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
                                ;WHICH SHOULD FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;(1)     "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
                                ;       VALUES AND TERMINATED WITH A CONTROL BYTE
                                ;(2)     "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
                                ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
                                ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
                                ;(3)     JSR     PC,GET
                                ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;       ERROR                   SUB DEFINES ERROR NUMBER
                                ;       ???
                                ;
037516  000240          GET:    NOP
037520  062716  000004          ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
037524  105076  000000          CLRB    @(SP)           ;ERROR CALL
037530  162716  000004          SUB     #4,(SP)
037534  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037536  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037540  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037542  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
037544  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
037546  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
037552  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
037556  013700  001276          MOV     $BASE,R0
037562  012702  001330          MOV     #GETBUF,R2
037566  012704  001506          MOV     #GETINX,R4
037572  012737  037700  000004  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
037600  012737  000300  000006  MOV     #PR6,ERRVEC+2
037606  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
037614  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
037622  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
037630  001007                  BNE     3$              ;YES!!
037632  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
037640  112776  000112  000016          MOVB    #112,@16(SP)    ;ERROR CALL
037646  000423                  BR      7$
037650  105714                  3$:     TSTB    (R4)            ;DONE??
037652  100433                  BMI     9$              ;YES!!
037654  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
037656  042701  177700          BIC     #^C<377>,R1     ;CLEAR ANY SIGN EXTENSION
037662  060001                  ADD     R0,R1
037664  112403                  MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
037666  042703  177700          BIC     #^C<377>,R3     ;CLEAR ANY SIGN EXTENSION
037672  060203                  ADD     R2,R3
037674  011113                  MOV     (R1),(R3)       ;READ REGISTER
```

```
8864  037676  000764                          BR      3$
8865
8866  037700  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
8867  037702  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
8868  037710  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
8869  037716  162766  000002  000016  7$:     SUB     #2,16(SP)
8870  037724  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
8871  037726  100405                          BMI     9$              ;YES!!
8872  037730  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
8873  037732  112403                          MOVB    (R4)+,R3        ;LOCATIONS
8874  037734  060203                          ADD     R2,R3
8875  037736  005013                          CLR     (R3)
8876  037740  000771                          BR      8$
8877  037742                          9$:
8878  037742  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8879  037746  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8880  037752  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
8881  037754  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8882  037756  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8883  037760  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8884  037762  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8885  037764  000207                          RTS     PC              ;RETURN
8886
```

```
                                .SBTTL  PUT SUBROUTINE

                                ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
                                ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
                                ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
                                ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
                                ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
                                ;FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;
                                ;(1)     "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
                                ;        OF REGISTERS TO BE WRITTEN.
                                ;(2)     "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
                                ;        REGISTER TO BE WRITTEN.
                                ;(3)     JSR     PC,PUT
                                ;        BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;        NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;        ERROR                   SUB DEFINES ERROR NUMBER
                                ;        ???
                                ;

037766  000240                  PUT:    NOP
037770  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037772  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037774  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037776  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
040000  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
040002  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
040006  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
040012  013700  001276                  MOV     $B-SE,R0
040016  012702  001400                  MOV     #PUTBUF,R2
040022  012704  001535                  MOV     #PUTINX,R4
040026  012737  040134  000004          MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
040034  012737  000300  000006          MOV     #PR6,ERRVEC+2
040042  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
040050  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
040056  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
040064  001007                          BNE     3$              ;YES!!
040066  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
040074  112776  000112  000016          MOVB    #112,@16(SP)    ;USER'S ERROR CALL
040102  000423                          BR      7$
040104  105714                  3$:     TSTB    (R4)            ;DONE??
040106  100424                          BMI     9$              ;YES!!
040110  111401                          MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
040112  042701  177700                  BIC     #^C<IDXMSK>,R1  ;CLEAR ANY SIGN EXTENSION
040116  060001                          ADD     R0,R1
040120  112403                          MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
040122  042703  177700                  BIC     #^C<IDXMSK>,R3  ;CLEAR ANY SIGN EXTENSION
040126  060203                          ADD     R2,R3
040130  011311                          MOV     (R3),(R1)       ;WRITE REGISTER
040132  000764                          BR      3$

040134  022626                  5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
040136  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
040144  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
040152  162766  000002  000016  7$:     SUB     #2,16(SP)
```

```
8943
8944   040160                                  9$:
8945   040160  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8946   040164  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8947   040170  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
8948   040172  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
8949   040174  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
8950   040176  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8951   040200  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8952   040202  000207                                  RTS     PC              ;RETURN
8953
```

```
8954                                        .SBTTL  SIZE CLOCK SUBROUTINE
8955
8956   040204                               SIZCLK:
8957   040204  013746  000004                       MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
8958   040210  013746  000006                       MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
8959   040214  012737  040252  000004               MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
8960   040222  012737  000300  000006               MOV     #PR6,ERRVEC+2
8961   040230  012737  177546  001502               MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
8962   040236  012737  000100  001504               MOV     #100,CLKVCT
8963   040244  005777  141232                       TST     @CLKADR         ;TEST FOR KW11-L PRESENT
8964   040250  000421                               BR      3$              ;YES - KW11-L IS PRESENT
8965   040252  022626                       1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
8966   040254  012737  040304  000004               MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
8967   040262  012737  172540  001502               MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
8968   040270  012737  000104  001504               MOV     #104,CLKVCT
8969   040276  005777  141200                       TST     @CLKADR         ;TEST FOR KW11-P PRESENT
8970   040302  000404                               BR      3$              ;YES - KW11-P IS PRESENT
8971   040304  022626                       2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
8972   040306  062766  000002  000004               ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
8973   040314                               3$:
8974   040314  012637  000006                       MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8975   040320  012637  000004                       MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8976   040324  000207                               RTS     PC              ;RETURN TO USER
```

```
8977                                    .SBTTL  TIMEOUT SUBROUTINE
8978                                    ;
8979                                    ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
8980                                    ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
8981                                    ;
8982                                    ;CALL:  JSR     PC,TIMOUT
8983                                    ;       ???                             RETURN HERE
8984
8985  040326                            TIMOUT:
8986  040326  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8987  040330  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8988  040332  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8989  040334  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
8990  040340  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
8991  040344  012737  040446  000004           MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
8992  040352  012737  000300  000006           MOV     #PR6,ERRVEC+2
8993  040360  013700  001276                   MOV     $BASE,R0        ;R0=RM03 ADDRESS
8994  040364  013701  001502                   MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
8995  040370  012702  000037                   MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
8996  040374  020127  172540           1$:     CMP     R1,#172540      ;KW11-P CLOCK??
8997  040400  001003                           BNE     2$              ;NO!!
8998  040402  012761  000001  000002           MOV     #1,2(R1)        ;SET COUNTER
8999  040410  012711  000005           2$:     MOV     #BIT2!BIT0,(R1) ;START COUNTER
9000
9001  040414  016046  000000           3$:     MOV     RMCS1(R0),-(SP) ;GET STATUS
9002  040420  042716  177576                   BIC     #^C<RDY!GO>,(SP)
9003  040424  022726  000200                   CMP     #RDY,(SP)+      ;RDY=1,GO=0??
9004  040430  001420                           BEQ     5$              ;YES!!
9005  040432  032711  000200                   BIT     #BIT7,(R1)      ;TIMER DONE??
9006  040436  001766                           BEQ     3$              ;NO!!
9007  040440  005302                           DEC     R2              ;DEC NUMBER OF CYCLES
9008  040442  001354                           BNE     1$              ;CONTINUE IF NOT DONE
9009  040444  000412                           BR      5$
9010  040446  022626           4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
9011  040450  062766  000004  000012           ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
9012  040456  112776  000007  000012           MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
9013  040464  162766  000002  000012           SUB     #2,12(SP)
9014
9015  040472                            5$:
9016  040472  012637  000006                   MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
9017  040476  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
9018  040502  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
9019  040504  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
9020  040506  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
9021  040510  000207                           RTS     PC              ;RETURN TO USER
9022
```

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

```
 9079  040600  162716  000002            MB      #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9080  040604  004736                    JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
 9081  040606  162716  000010            SUB     #10.,(SP)        ;RESTORE (SP)
 9082  040612  000240                    NOP
 9083  040614  000137  041334            JMP     10$              ;SKIP OTHER CHECKS
 9084  040620                        1$:
 9085
 9086                                    ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
 9087                                    ;THE DEVICE IS NONEXISTANT
 9088  040620  032737  004000  001330            BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
 9089  040626  001045                            BNE     5$               ;YES!!
 9090  040630  013737  001330  001140            MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9091  040636  052737  004000  001140            BIS     #DVA,$GDDAT
 9092  040644  013737  001330  001142            MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9093  040652  062716  000004                    ADD     #4,(SP)
 9094  040656  112776  000002  000000            MOVB    #2,@(SP)         ;ERROR #2
 9095  040664  032737  010000  001340            BIT     #NED,RMCS2I      ;WAS NED SET??
 9096  040672  001414                            BEQ     2$               ;NO!!
 9097  040674  013737  001340  001140            MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
 9098  040702  013737  001340  001142            MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
 9099  040710  042737  010000  001140            BIC     #NED,$GDDAT
 9100  040716  112776  000003  000000            MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
 9101  040724  162716  000002                2$: SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9102  040730  004736                            JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
 9103  040732  162716  000010                    SUB     #10.,(SP)        ;RESTORE (SP)
 9104  040736  000240                            NOP
 9105  040740  000575                            BR      10$              ;SKIP OTHER CHECKS
 9106  040742                            5$:
 9107
 9108                                    ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
 9109  040742  032737  000200  001330            BIT     #RDY,RMCS1I      ;CONTROLLER READY??
 9110  040750  001030                            BNE     7$               ;YES!!
 9111  040752  013737  001330  001140            MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9112  040760  052737  000200  001140            BIS     #RDY,$GDDAT
 9113  040766  042737  160001  001140            BIC     #SC!TRE!MCPE!GO,$GDDAT
 9114  040774  013737  001330  001142            MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9115  041002  062716  000004                    ADD     #4,(SP)
 9116  041006  112776  000004  000000            MOVB    #4,@(SP)         ;ERROR #4
 9117  041014  162716  000002                    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9118  041020  004736                            JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
 9119  041022  162716  000010                    SUB     #10.,(SP)        ;RESTORE (SP)
 9120  041026  000240                            NOP
 9121  041030  000541                            BR      10$              ;SKIP OTHER CHECKS
 9122  041032                            7$:
 9123
 9124                                    ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
 9125  041032  032737  000001  001330            BIT     #GO,RMCS1I       ;GO RESET??
 9126  041040  001431                            BEQ     8$               ;YES!!
 9127  041042  032737  000200  001342            BIT     #DRY,RMDSI       ;DRIVE READY??
 9128  041050  001025                            BNE     8$               ;YES!!
 9129  041052  013737  001330  001140            MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9130  041060  042737  160001  001140            BIC     #SC!TRE!MCPE!GO,$GDDAT
 9131  041066  013737  001330  001142            MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9132  041074  062716  000004                    ADD     #4,(SP)
 9133  041100  112776  000005  000000            MOVB    #5,@(SP)         ;ERROR #5
 9134  041106  162716  000002                    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
 9135  041112  004736                        JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
 9136  041114  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
 9137  041120  000240                        NOP
 9138  041122  000504                        BR      10$
 9139  041124                          8$:
 9140
 9141                                  ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
 9142                                  ;PARITY ON THE MASSBUS CONTROL BUS
 9143  041124  032737  020000  001330        BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
 9144  041132  001425                        BEQ     9$              ;NO!!
 9145  041134  013737  001330  001140        MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
 9146  041142  042737  160001  001140        BIC     #SC!TRE!MCPE!GO,$GDDAT
 9147  041150  013737  001330  001142        MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
 9148  041156  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
 9149  041162  112776  000013  000000        MOVB    #13,@(SP)       ;ERROR #47
 9150  041170  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9151  041174  004736                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9152  041176  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
 9153  041202  000240                        NOP
 9154  041204  000453                        BR      10$
 9155  041206                          9$:
 9156
 9157                                  ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
 9158  041206  032737  000010  001344        BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
 9159  041214  001451                        BEQ     11$             ;NO!!
 9160  041216  032737  000010  001372        BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
 9161  041224  001045                        BNE     11$             ;NOT SURE!!
 9162  041226  032737  000010  001414        BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
 9163  041234  001413                        BEQ     93$             ;NO!!
 9164  041236  010046                        MOV     R0,-(SP)        ;PUSH R0 ON STACK
 9165  041240  012700  001535                MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
 9166  041244  122710  000014          91$:  CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
 9167  041250  001002                        BNE     92$
 9168  041252  012600                        MOV     (SP)+,R0        ;POP STACK INTO R0
 9169  041254  000431                        BR      11$             ;PAR WAS SET BY TEST
 9170  041256  105720                  92$:  TSTB    (R0)+           ;END OF TABLE??
 9171  041260  100371                        BPL     91$             ;NO!!
 9172  041262  012600                        MOV     (SP)+,R0        ;POP STACK INTO R0
 9173  041264  013737  001344  001140  93$:  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 9174  041272  042737  000010  001140        BIC     #PAR,$GDDAT
 9175  041300  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 9176  041306  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 9177  041312  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
 9178  041320  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9179  041324  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR
 9180  041326  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 9181  041332  000240                        NOP
 9182  041334  062716  000010          10$:  ADD     #10,(SP)                ;RETURN TO ERROR
 9183  041340  000240                  11$:  NOP                     ;RETURN TO NO ERROR
 9184  041342  000207                        RTS     PC
 9185
```

```
                                        .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
                                        ;
                                        ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
                                        ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
                                        ;CONTENTS.  THESE ERRORS ARE DEEMED
                                        ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
                                        ;BEING PERFORMED.  WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
                                        ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
                                        ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL.  AFTER THE TEST ROUTINE
                                        ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
                                        ;OTHER ERRORS.  WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
                                        ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
                                        ;
                                        ;CALL:  JSR     PC,SECERR
                                        ;       BR      ???             RETURN HERE IF NO ERROR
                                        ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                        ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                        ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                        ;       ???                     RETURN HERE IF NO MORE ERRORS
                                        ;
                                        ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
                                        ;INPUT REGISTER BUFFER.
                                        ;
041344                                  SECERR:
                                        ;
                                        ;;*****************************************************************
                                        ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
041344  013737  001400  045204                MOV     RMCS10,$15$     ;STORE FUNCTION CODE
041352  042737  177701  045204                BIC     #^C<F0!F1!F2!F3!F4>,$15$
041360  062716  000004                        ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
041364  105076  000000                        CLRB    @(SP)           ;CLEAR ERROR NUMBER
041370  162716  000004                        SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                                        ;
                                        ;;*****************************************************************
                                        ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
                                        ;
                                        ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
041374  032737  000200  001342                BIT     #DRY,RMDSI      ;DRIVE READY??
041402  001024                                BNE     5$              ;YES!!
041404  013737  001342  001142                MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
041412  042737  177577  001142                BIC     #^CDRY,$BDDAT
041420  012737  000200  001140                MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
041426  062716  000004                        ADD     #4,(SP)
041432  112776  000010  000000                MOVB    #10,@(SP)       ;ERROR NUMBER
041440  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
041444  004736                                JSR     PC,@(SP)+       ;REPORT NOT READY
041446  162716  000010                        SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
041452  000240                                NOP
                                        ;
                                        ;REPORT ERROR IF GO BIT IS NOT RESET
041454  032737  000001  001330  5$:           BIT     #GO,RMCS1I      ;GO BIT RESET??
041462  001423                                BEQ     10$             ;YES!!
041464  013737  001330  001142                MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
041472  042737  177776  001142                BIC     #^CGO,$BDDAT
041500  005037  001140                        CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
041504  062716  000004                        ADD     #4,(SP)
```

```
9242  041510  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
9243  041516  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9244  041522  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
9245  041524  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9246  041530  000240                          NOP
9247
9248                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
9249  041532  013737  001330  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
9250  041540  042737  177701  001142          BIC     #^C76,$BDDAT
9251  041546  013737  045204  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
9252  041554  023737  001142  001140          CMP     $BDDAT,$GDDAT
9253  041562  001413                          BEQ     15$             ;YES!!
9254  041564  062716  000004                  ADD     #4,(SP)
9255  041570  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
9256  041576  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9257  041602  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
9258  041604  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9259  041610  000240                          NOP
9260  041612                          15$:
9261                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
9262                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
9263                                  ;OTHER ERRORS ARE SET
9264  041612  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
9265  041616  005737  001344                  TST     RMER1I          ;IS RMER1 =0??
9266  041622  001003                          BNE     20$             ;NO!!
9267  041624  005737  001372                  TST     RMER2I          ;IS RMER2=0??
9268  041630  001403                          BEQ     25$             ;YES!!
9269  041632  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
9270  041640  013737  001342  001142  25$:     MOV     RMDSI,$BDDAT
9271  041646  042737  137777  001142          BIC     #^CERR,$BDDAT
9272  041654  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
9273  041662  001412                          BEQ     30$             ;YES!!
9274  041664  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
9275  041670  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
9276  041676  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9277  041702  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
9278  041704  162716  000010                  SUB     #10,(SP)
9279
9280                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
9281                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
9282                                  ;SET TRE IS SET
9283  041710  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
9284  041714  013746  001340                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
9285  041720  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
9286  041724  001010                          BNE     35$             ;YES!!
9287  041726  032737  040000  001342          BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
9288  041734  001407                          BEQ     40$             ;NO!!
9289  041736  022737  000030  045204          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
9290  041744  103003                          BHIS    40$             ;NO!!
9291  041746  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
9292  041754  013737  001330  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
9293  041762  042737  137777  001142          BIC     #^CTRE,$BDDAT
9294  041770  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
9295  041776  001413                          BEQ     45$             ;YES!!
9296  042000  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
9297  042004  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
 9298  042012  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9299  042016  004736                    JSR     PC,@(SP)+        ;REPORT TRE ERROR
 9300  042020  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
 9301  042024  000240                    NOP
 9302  042026                        45$:
 9303
 9304                                  ;;*****************************************************************
 9305                                  ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
 9306                                  ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
 9307                                  ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
 9308                                  ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
 9309                                  ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
 9310                                  ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
 9311
 9312                                  ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
 9313  042026  010046                    MOV     R0,-(SP)         ;;PUSH R0 ON STACK
 9314  042030  013700  045204            MOV     51$,R0           ;R0 = FUNCTION CODE
 9315  042034  016037  064674  045176    MOV     FNCDTB(R0),50O$  ;STORE ENTRY
 9316  042042  012600                    MOV     (SP)+,R0         ;;POP STACK INTO R0
 9317
 9318                                  ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
 9319                                  ;ATA IS NOT SET AND SHOULD BE SET.
 9320  042044  013737  045176  001140    MOV     50O$,$GDDAT      ;GET EXPECTED ATA STATUS
 9321  042052  032737  040000  001342    BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
 9322  042060  001403                    BEQ     50$              ;NO !!
 9323  042062  052737  100000  001140    BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
 9324  042070  042737  077777  001140  50$:    BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
 9325  042076  013737  001342  001142    MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
 9326  042104  042737  077777  001142    BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
 9327  042112  023737  001140  001142    CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
 9328  042120  001413                    BEQ     55$              ;YES !!
 9329  042122  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9330  042126  112776  000006  000000    MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
 9331  042134  162716  000002            SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9332  042140  004736                    JSR     PC,@(SP)+        ;REPORT ERROR
 9333  042142  162716  000010            SUB     #10,(SP)         ;RESTORE SP
 9334  042146  000240                    NOP
 9335  042150                        55$:
 9336
 9337                                  ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
 9338                                  ;WITH FUNCTION CODE TABLE
 9339  042150  013737  045176  001140    MOV     50O$,$GDDAT      ;GET EXPECTED ILF
 9340  042156  042737  177776  001140    BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
 9341  042164  013737  001344  001142    MOV     RMER1I,$BDDAT    ;GET RECEIVED ILF
 9342  042172  042737  177776  001142    BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
 9343  042200  023737  001140  001142    CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
 9344  042206  001412                    BEQ     60$              ;YES !!
 9345  042210  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9346  042214  112776  000254  000000    MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9347  042222  162716  000002            SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9348  042226  004736                    JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
 9349  042230  162716  000010            SUB     #10,(SP)         ;MOVE SP TO NO ERROR
 9350  042234  005037  001140      60$:  CLR     $GDDAT           ;CLEAR EXPECTED STATUS
 9351
 9352                                  ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
 9353  042240  013746  045176            MOV     50O$,-(SP)       ;GET WCE STATUS ENABLE
```

```
 9354  042244  052716  137777                BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
 9355  042250  013737  001340  001142        MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
 9356  042256  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
 9357  042262  001412                        BEQ     90$             ;BRANCH IF WCE OK
 9358  042264  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9359  042270  112776  000026  000000        MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
 9360  042276  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9361  042302  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9362  042304  162716  000010                SUB     #10,(SP)        ;RESTORE ERROR
 9363  042310                        90$:
 9364
 9365                                ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
 9366  042310  013746  045176                MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
 9367  042314  052716  157777                BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
 9368  042320  013737  001344  001142        MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9369  042326  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
 9370  042332  001412                        BEQ     100$            ;BRANCH IF OPI OK
 9371  042334  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9372  042340  112776  000164  000000        MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
 9373  042346  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9374  042352  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9375  042354  162716  000010                SUB     #10,(SP)        ;RESTORE SP
 9376  042360                        100$:
 9377
 9378                                ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
 9379                                ;SET AND VV IS NOT RESET
 9380  042360  013746  045176                MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
 9381  042364  032737  000100  001342        BIT     #VV,RMDSI       ;IS VV SET
 9382  042372  001402                        BEQ     105$            ;NO !!
 9383  042374  042716  010000                BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
 9384  042400  052716  167777        105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
 9385  042404  013737  001372  001142        MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
 9386  042412  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
 9387  042416  001412                        BEQ     110$            ;BRANCH IF IVC OK
 9388  042420  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9389  042424  112776  000165  000000        MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
 9390  042432  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9391  042436  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9392  042440  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 9393  042444                        110$:
 9394
 9395                                ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
 9396                                ; ALL WRITE ERRORS, I.E.,
 9397                                ;       RMER1 - WLE, WCF
 9398                                ;       RMER2 - DPE
 9399                                ;       RMCS2 - UPE.
 9400                                ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
 9401                                ;WRITE ERROR ENABLE BIT IS RESET.
 9402
 9403                                ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
 9404                                ;THE DRIVE IS NOT WRITE PROTECTED
 9405  042444  012746  177777                MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
 9406  042450  032737  004000  045176        BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
 9407  042456  001404                        BEQ     115$            ;NO !!
 9408  042460  032737  004000  001342        BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
 9409  042466  001002                        BNE     120$            ;YES !!
```

```
9410  042470  042716  004000          115$:   BIC     #WLE,(SP)          ;RESET WLE ENABLE
9411  042474  013737  001344  001142  120$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
9412  042502  042637  001142                  BIC     (SP)+,$BDDAT       ;CLEAR WLE IF ENABLED
9413  042506  001412                          BEQ     125$               ;BRANCH IF WLE OK
9414  042510  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9415  042514  112776  000023  000000          MOVB    #23,@(SP)          ;WRITE ERROR NUMBER IN CALL
9416  042522  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9417  042526  004736                          JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
9418  042530  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9419  042534                          125$:
9420
9421                                  ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
9422  042534  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ENABLED
9423  042540  032737  004000  045176          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9424  042546  001002                          BNE     130$               ;YES !!
9425  042550  042716  000040                  BIC     #WCF,(SP)          ;DISABLE WCF ERROR
9426  042554  013737  001344  001142  130$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
9427  042562  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET WCF IF ENABLED
9428  042566  001412                          BEQ     135$               ;BRANCH IF WCF OK
9429  042570  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9430  042574  112776  000025  000000          MOVB    #25,@(SP)          ;WRITE ERROR NUMBER IN CALL
9431  042602  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9432  042606  004736                          JSR     PC,@(SP)+          ;REPORT ERROR
9433  042610  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9434  042614                          135$:
9435
9436                                  ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9437  042614  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
9438  042620  032737  004000  045176          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9439  042626  001002                          BNE     140$               ;YES !!
9440  042630  042716  000010                  BIC     #DPE,(SP)          ;RESET DPE ENABLE
9441  042634  013737  001372  001142  140$:   MOV     RMER2I,$BDDAT      ;GET RECEIVED STATUS
9442  042642  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET DPE IF ENABLED
9443  042646  001412                          BEQ     1  $               ;BRANCH IF DPE OK
9444  042650  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9445  042654  112776  000040  000000          MOVB    #40,@(SP)          ;WRITE ERROR NUMBER IN CALL
9446  042662  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9447  042666  004736                          JSR     PC,@(SP)+          ;REPORT ERROR
9448  042670  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9449  042674                          145$:
9450
9451                                  ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9452  042674  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
9453  042700  032737  004000  045176          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9454  042706  001002                          BNE     150$               ;YES !!
9455  042710  042716  020000                  BIC     #UPE,(SP)          ;DISABLE UPE ERROR
9456  042714  013737  001340  001142  150$:   MOV     RMCS2I,$BDDAT      ;GET RECEIVED STATUS
9457  042722  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET UPE IF ENABLED
9458  042726  001412                          BEQ     155$               ;BRANCH IF UPE OK
9459  042730  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9460  042734  112776  000024  000000          MOVB    #24,@(SP)          ;WRITE ERROR NUMBER IN CALL
9461  042742  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9462  042746  004736                          JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
9463  042750  162716  000010                  SUB     #10,(SP)           ;MOVE SP TO NO ERROR
9464  042754                          155$:
9465
```

```
                                        ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
042754  013746  045176                          MOV     500$,-(SP)      ;GET IAE ENABLE
042760  052716  175777                          BIS     #^C<IAE>,(SP)   ;SET ALL OTHER BITS
042764  013737  001344  001142                  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
042772  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
042776  001412                                  BEQ     160$            ;BRANCH IF IAE IS OK
043000  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043004  112776  000166  000000                  MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
043012  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043016  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043020  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
043024                                  160$:
                                        ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                                        ; ALL READ/WRITE ERRORS, I.E.,
                                        ;       RMCS1 - TRE
                                        ;       RMCS2 - DLT,NEM,MXF
                                        ;       RMDS  - LBT
                                        ;       RMER1 - AOE
                                        ;NOTE:
                                        ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
                                        ;CYLINDER REGISTER IS WRITTEN
                                        ;NOTE:
                                        ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
                                        ;NOTE:
                                        ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE

                                        ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
043024  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
043030  032737  001000  045176                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
043036  001002                                  BNE     165$            ;YES !!
043040  042716  100000                          BIC     #DLT,(SP)       ;RESET DLT ENABLE
043044  013737  001340  001142          165$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
043052  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
043056  001412                                  BEQ     170$            ;BRANCH IF DLT IS OK
043060  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043064  112776  000032  000000                  MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
043072  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043076  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043100  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
043104                                  170$:

                                        ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
043104  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
043110  032737  001000  045176                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
043116  001002                                  BNE     175$            ;YES !!
043120  042716  004000                          BIC     #NEM,(SP)       ;DISABLE NEM
043124  013737  001340  001142          175$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
043132  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
043136  001412                                  BEQ     180$            ;BRANCH IF NEM IS OK
043140  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043144  112776  000167  000000                  MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
043152  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043156  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043160  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
9522  043164                                  180$.
9523
9524                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
9525  043164  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
9526  043170  032737  001000  045176          BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
9527  043176  001002                          BNE     185$            ;YES !!
9528  043200  042716  001000                  BIC     #MXF,(SP)       ;DISABLE MXF ERROR
9529  043204  013737  001340  001142  185$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
9530  043212  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
9531  043216  001412                          BEQ     190$            ;BRANCH IF MXF IS OK
9532  043220  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9533  043224  112776  000033  000000          MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
9534  043232  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9535  043236  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9536  043240  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9537  043244                                  190$:
9538
9539                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
9540  043244  012746  177777                  MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
9541  043250  032737  001000  045176          BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
9542  043256  001404                          BEQ     191$            ;NO !!
9543  043260  032737  002000  001342          BIT     #LBT,RMOSI      ;IS LBT ALSO SET ??
9544  043266  001002                          BNE     195$            ;YES !!
9545  043270  042716  001000          191$:   BIC     #AOE,(SP)       ;DISABLE AOE
9546  043274  013737  001344  001142  195$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9547  043302  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
9548  043306  001412                          BEQ     200$            ;BRANCH IF AOE IS OK
9549  043310  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9550  043314  112776  000020  000000          MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
9551  043322  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9552  043326  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9553  043330  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9554  043334                                  200$:
9555
9556                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
9557                                          ;HEADER ERRORS, I.E.,
9558                                          ;       RMER1 - HCRC,HCE,FER
9559                                          ;       RMER2 - BSE
9560
9561                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
9562  043334  032737  002000  001362          BIT     #HCI,RMOFI      ;IS HCI SET ??
9563  043342  001403                          BEQ     201$            ;NO !!
9564  043344  042737  000200  045176          BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
9565  043352                                  201$:
9566
9567                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
9568  043352  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9569  043356  032737  000200  045176          BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
9570  043364  001002                          BNE     205$            ;YES !!
9571  043366  042716  000400                  BIC     #HCRC,(SP)      ;DISABLE HCRC
9572  043372  013737  001344  001142  205$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9573  043400  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
9574  043404  001412                          BEQ     210$            ;BRANCH IF HCRC IS OK
9575  043406  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9576  043412  112776  000035  000000          MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
9577  043420  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
 9578  043424  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9579  043426  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9580  043432                        210$:
 9581
 9582                                ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
 9583  043432  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
 9584  043436  032737  000200  045176        BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
 9585  043444  001002                        BNE     215$            ;YES !!
 9586  043446  042716  000200                BIC     #HCE,(SP)       ;DISABLE HCE
 9587  043452  013737  001344  001142  215$: MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9588  043460  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
 9589  043464  001412                        BEQ     220$            ;BRANCH IF HCE IS OK
 9590  043466  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9591  043472  112776  000036  000000        MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9592  043500  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9593  043504  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9594  043506  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9595  043512                        220$:
 9596
 9597                                ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
 9598  043512  012746  177777                MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
 9599  043516  032737  000200  045176        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
 9600  043524  001002                        BNE     225$            ;YES !!
 9601  043526  042716  000020                BIC     #FER,(SP)       ;DISABLE FER
 9602  043532  013737  001344  001142  225$: MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9603  043540  042637  001142                BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
 9604  043544  001412                        BEQ     230$            ;BRANCH IF FER OK
 9605  043546  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9606  043552  112776  000037  000000        MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9607  043560  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9608  043564  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9609  043566  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9610  043572                        230$:
 9611
 9612                                ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
 9613  043572  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
 9614  043576  032737  000200  045176        BIT     #HCE,500$       ;ARE THEY ENABLED ??
 9615  043604  001002                        BNE     235$            ;YES !!
 9616  043606  042716  100000                BIC     #BSE,(SP)       ;DISABLE BSE
 9617  043612  013737  001372  001142  235$: MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
 9618  043620  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
 9619  043624  001412                        BEQ     240$            ;BRANCH IF BSE OK
 9620  043626  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9621  043632  112776  000354  000000        MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
 9622  043640  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9623  043644  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9624  043646  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9625  043652                        240$:
 9626
 9627                                ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
 9628                                ;FIELD ERRORS, I.E.,
 9629                                ;       RMCS2 - MDPE
 9630                                ;       RMER1 - DCK,ECH
 9631                                ;NOTE:
 9632                                ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
 9633                                ;DCK IS SET.
```

```
                                        ;REPORT ERROR IF MOPE IS SET AND IS NOT ENABLED
043652  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ENABLED
043656  032737  000100  045176          BIT     #ECH,$OOS       ;ARE DATA FIELD ERRORS ENABLED ??
043664  001002                          BNE     245$            ;YES !!
043666  042716  000400                  BIC     #MOPE,(SP)      ;DISBALE MOPE
043672  013737  001340  001142  245$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
043700  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR MOPE IF ENABLED
043704  001412                          BEQ     250$            ;BRANCH IF MOPE OK
043706  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043712  112776  000027  000000          MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
043720  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043724  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043726  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
043732                          250$:

                                        ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
043732  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ENABLED
043736  032737  000100  045176          BIT     #ECH,$OOS       ;ARE THEY ENABLED ??
043744  001002                          BNE     255$            ;YES !!
043746  042716  100000                  BIC     #DCK,(SP)       ;DISABLE DCK
043752  013737  001344  001142  255$:   MOV     RMERII,$BDDAT   ;GET RECEIVED STATUS
043760  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
043764  001412                          BEQ     260$            ;BRANCH IF DCK IS OK
043766  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043772  112776  000030  000000          MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
044000  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
044004  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
044006  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
044012                          260$:

                                        ;REPORT ERROR IF ECH IS SET AND,
                                        ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
                                        ;       ECI IS SET, OR
                                        ;       DCK IS NOT SET,
044012  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ENABLED
044016  032737  000100  045176          BIT     #ECH,$OOS       ;ARE ERRORS ENABLED ??
044024  001410                          BEQ     265$            ;NO !!
044026  032737  004000  001362          BIT     #ECI,RMOFI      ;IS ECI SET ??
044034  001004                          BNE     265$            ;YES !!
044036  032737  100000  001344          BIT     #DCK,RMERII     ;IS DCK ALSO SET ??
044044  001002                          BNE     270$            ;YES !!
044046  042716  000100          265$:   BIC     #ECH,(SP)       ;DISABLE ECH
044052  013737  001344  001142  270$:   MOV     RMERII,$BDDAT   ;GET RECEIVED STATUS
044060  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
044064  001412                          BEQ     275$            ;BRANCH IF ECH IS OK
044066  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
044072  112776  000031  000000          MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
044100  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
044104  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
044106  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
044112                          275$:

                                        ;;*****************************************************************
                                        ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
```

```
 9690  044112  022737  000030  045204          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9691  044120  103402                          BLO     280$            ;YES!!
 9692  044122  000137  045150                  JMP     355$
 9693
```

```
                                        ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
                                        ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
044126  013737  001332  001142  280$:   MOV     RMWCI,$BODAT    ;WORD COUNT ZERO??
044134  001421                          BEQ     28?$            ;YES
044136  032737  040000  001330          BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
044144  001015                          BNE     285$            ;YES!!
044146  062716  000004                  ADD     #4,(SP)
044152  112776  000015  000000          MOVB    #15,3(SP)       ;ERROR NUMBER
044160  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
044164  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
044170  004736                          JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
044172  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
044176  000240                          NOP
                                        ;REPORT ERROR IF RMBA IS NOT CORRECT
044200  013737  001332  001140  285$:   MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
044206  163737  001402  001140          SUB     RMWCO,$GDDAT
044214  006337  001140                  ASL     $GDDAT
044220  063737  001404  001140          ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
044226  032737  00 '0  001340          BIT     #BAI,RMCS2I     ;WAS BAI SET ??
044234  001403                          BEQ     29U$            ;NO !!
044236  013737  0014L   001140          MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
044244  023737  001140  001334  290$:   CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
044252  001416                          BEQ     295$            ;YES!!
044254  013737  001334  001142          MOV     RMBAI,$BODAT    ;BAD DATA FOR TYPEOUT
044262  062716  000004                  ADD     #4,(SP)
044266  112776  000016  000000          MOVB    #16,3(SP)       ;ERROR NUMBER
044274  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
044300  004736                          JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
044302  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
044306  000240                          NOP
                                        ;COMPUTE NUMBER OF SECTORS TRANSFERRED
044310  005046                  295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
044312  013746  001332                  MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
044316  163716  001402                  SUB     RMWCO,(SP)
044322  012746  000400                  MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
044326  032737  000002  001400          BIT     #BIT1,RMCS1O    ;HEADER & DATA COMMAND ??
044334  001402                          BEQ     300$            ;NO !!
044336  012716  000402                  MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
044342  005266  000004          300$:   INC     4(SP)           ;INCREMENT SECTOR COUNT
044346  161666  000002                  SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
044352  003373                          BGT     300$            ;CONTINUE IF NOT DONE
044354  022626                          CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
                                        ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
044356  013737  001406  045202          MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
044364  013737  001406  045200          MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
044372  013737  001434  045176          MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
044400  042737  177740  045202          BIC     #^C37,510$
044406  000337  045200                  SWAB    505$
044412  042737  177770  045200          BIC     #^C7,505$
044420  062637  045202                  ADD     (SP)+,510$
044424  023727  045202  000040  305$:   CMP     510$,#32.       ;SECTOR OVEFLOWED??
044432  103420                          BLO     315$            ;NO!!
044434  023727  045202  000240          CMP     510$,#(5*32.)   ;CYLINDERS WORTH??
044442  103406                          BLO     310$            ;NO!!
044444  005237  045176                  INC     500$            ;YES INCREMENT CYLINDER
```

```
9750  044450  162737  000240  045202          SUB     #<5*32.>,510$   ;ADJUST SECTOR
9751  044456  000762                          BR      305$            ;TRY AGAIN
9752  044460  005237  045200          310$:   INC     505$            ;INCREMENT TRACK
9753  044464  162737  000040  045202          SUB     #32.,510$       ;ADJUST SECTOR
9754  044472  000754                          BR      305$            ;TRY AGAIN
9755
9756  044474  023727  045200  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
9757  044502  103406                          BLO     320$            ;NO!!
9758  044504  005237  045176                  INC     500$            ;INCREMENT CYLINDER
9759  044510  162737  000005  045200          SUB     #5,505$ ;ADJUST TRACK
9760  044516  000766                          BR      315$            ;TRY AGAIN
9761                          ;REPORT ERROR IF "LBT" IS NOT CORRECT
9762  044520  005037  001140          320$:   CLR     $GOODAT         ;SET GOOD DATA FOR LBT=0
9763  044524  022737  001467  045176          CMP     #823.,500$      ;SHOULD LBT BE SET??
9764  044532  101007                          BHI     325$            ;NO!!
9765  044534  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS IAE SET ??
9766  044542  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
9767  044546  012737  002000  001140          MOV     #LBT,$GOODAT    ;SET GOOD DATA FOR LBT=1
9768  044552  013737  001342  001142  325$:   MOV     RMDS1,$BADAT    ;BAD DATA FOR TYPEOUT
9769  044560  042737  175777  001142          BIC     #^CLBT,$BADAT
9770  044566  023737  001140  001142          CMP     $GOODAT,$BADAT  ;IS LBT CORRECT??
9771  044574  001413                          BEQ     330$            ;YES!!
9772  044576  062716  000004                  ADD     #4,(SP)
9773  044602  112776  000017  000000          MOVB    #17,2(SP)       ;ERROR NUMBER
9774  044610  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9775  044614  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
9776  044616  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9777  044622  000240                          NOP
9778                          ;REPORT ERROR IF "AOE" IS INCORRECT
9779  044624  005037  001140          330$:   CLR     $GOODAT         ;SET FOR AOE=0
9780  044630  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
9781  044636  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
9782  044640  022737  001467  045176          CMP     #823.,500$      ;SHOULD AOE BE SET??
9783  044646  101025                          BHI     340$            ;NO!!
9784  044650  005737  045200                  TST     505$            ;MAYBE
9785  044654  001012                          BNE     335$            ;YES
9786  044656  005737  045202                  TST     510$
9787  044662  001007                          BNE     335$            ;YES !!
9788  044664  032737  000010  045204          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
9789  044672  001413                          BEQ     340$            ;NO !!
9790  044674  005737  001332                  TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
9791  044700  001410                          BEQ     340$            ;YES !!
9792  044702  012737  001000  001140  335$:   MOV     #AOE,$GOODAT    ;SET FOR AOE=1
9793  044710  005037  045200                  CLR     505$            ;CLEAR EXPECTED TRACK
9794  044714  012737  000001  045202          MOV     #1,510$ ;EXPECT SECTOR=1
9795  044722  013737  001344  001142  340$:   MOV     RMER1I,$BADAT   ;BAD DATA FOR TYPEOUT
9796  044730  042737  176777  001142          BIC     #^CAOE,$BADAT
9797  044736  023737  001140  001142          CMP     $GOODAT,$BADAT  ;IS AOE CORRECTY??
9798  044744  001413                          BEQ     345$            ;YES!!
9799  044746  062716  000004                  ADD     #4,(SP)
9800  044752  112776  000020  000000          MOVB    #20,2(SP)       ;ERROR NUMBER
9801  044760  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9802  044764  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
9803  044766  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9804  044772  000240                          NOP
5805                          ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
9806  044774  032737  002000  001344  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
9807  045002  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
9808  045004  013737  045200  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
9809  045012  000337  001140                  SWAB    $GDDAT
9810  045016  113737  045202  001140          MOVB    510$,$GDDAT
9811  045024  013737  001336  001142          MOV     RMDA1,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
9812  045032  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
9813  045040  001413                          BEQ     350$            ;BRANCH IF EQUAL
9814  045042  062716  000004                  ADD     #4,(SP)
9815  045046  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
9816  045054  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9817  045060  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
9818  045062  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9819  045066  000240                          NOP
9820                                ;REPORT ERROR IF RMDC IS INCORRECT
9821  045070  013737  045176  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
9822  045076  042737  176000  001140          BIC     #^C1777,$GDDAT
9823  045104  013737  001364  001142          MOV     RMDC1,$BDDAT    ;SETUP RECEIVED CYLINDER
9824  045112  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
9825  045120  001413                          BEQ     355$            ;BRANCH IF EQUAL
9826  045122  062716  000004                  ADD     #4,(SP)
9827  045126  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
9828  045134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9829  045140  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
9830  045142  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9831  045146  000240                          NOP
```

```
 9832  045150  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
 9833  045154  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
 9834  045160  001403                          BEQ     360$
 9835  045162  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
 9836  045166  000402                          BR      365$
 9837  045170  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
 9838  045174  000207                  365$:   RTS     PC
 9839
 9840  045176  000000                  500$:   .WORD   0               ;CYLINDER
 9841  045200  000000                  505$:   .WORD   0               ;TRACK
 9842  045202  000000                  510$:   .WORD   0               ;SECTOR
 9843  045204  000000                  515$:   .WORD   0               ;FUNCTION CODE
 9844
 9845
```

```
                                        .SBTTL  DEVICE SELECT SUBROUTINE

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

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

045206                                  DEVSEL:

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

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

045356                                  20$:

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

045402                                  35$:


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

```
 9902  045406  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 9903  045412  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 9904  045416  000207                        RTS     PC              ;;EXIT
```

```
 9905                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
 9906                                   ;
 9907                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
 9908                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
 9909                                   ;
 9910                                   ;
 9911                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
 9912                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
 9913                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
 9914                                   ;
 9915                                   ;CALL:
 9916                                   ;(1)    JSR     PC,SEKSTS
 9917                                   ;       BR      ???             RETURN HERE IF NO ERROR
 9918                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
 9919                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
 9920                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
 9921                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
 9922                                   ;
 9923   045420                          SEKSTS:
 9924                                   ;
 9925                                   ;CLEAR USER'S ERROP  CALL
 9926   045420  000240                          NOP
 9927   045422  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
 9928   045426  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
 9929   045432  162716  000004                SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
 9930   045436  005037  046656                CLR     300$            ;CLEAR ERROR FLAGS
 9931
 9932                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
 9933                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "OPE" = 0
 9934   045442  032737  000010  001344        BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
 9935   045450  001424                        BEQ     1$              ;NO!!
 9936   045452  032737  000010  001372        BIT     #OPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
 9937   045460  001020                        BNE     1$              ;NOT SURE!!
 9938
 9939                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
 9940   045462  005037  001140                CLR     $GDDAT          ;EXPECTED STATUS
 9941   045466  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 9942   045474  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
 9943   045500  112776  000050  000000        MOVB    #50,@(SP)       ;ERROR #50
 9944   045506  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9945   045512  004736                        JSR     PC,@(SP)+
 9946   045514  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
 9947   045520  000437                        BR      3$              ;IAE SHOULD  BE ZERO
 9948
 9949
 9950                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
 9951                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
 9952   045522  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
 9953   045530  052737  040000  046656        BIS     #SKI,300$       ;SET SKI ERROR FLAG
 9954   045536  023727  001466  001466        CMP     RMDC0,#822.     ;CYLINDER > 822??
 9955   045544  101025                        BHI     3$              ;YES!!
 9956   045546  042737  040000  046656        BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
 9957   045554  123727  001406  000037        CMPB    RMDA0,#31.      ;SECTOR > 31??
 9958   045562  101016                        BHI     3$              ;YES!!
 9959   045564  123727  001406  000035        CMPB    RMDA0,#29.      ;SECTOR > 29 ??
 9960   045572  101412                        BLOS    2$              ;NO!!
```

```
 9961  045574  032737  010000  001432          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
 9962  045602  001406                          BEQ     3$              ;YES!!
 9963  045604  123727  001407  000004  2$:     CMPB    RMDAO+1,#4.     ;TRACK  >4??
 9964  045612  101002                          BHI     3$              ;YES!!
 9965  045614  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
 9966
 9967                                          ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
 9968  045620  013737  001344  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
 9969  045626  042737  175777  001142          BIC     #^CIAE,$BDDAT
 9970  045634  023737  001140  001142          CMP     $GDDAT,$BDDAT
 9971  045642  001004                          BNE     35$             ;IAE IN ERROR
 9972  045644  042737  040000  046656          BIC     #SKI,300$       ;CLEAR SKI FLAG
 9973  045652  000413                          BR      5$              ;GO CHECK NEXT ERROR
 9974  045654                                  35$:
 9975                                          ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
 9976  045654  062716  000004                  ADD     #4,(SP)
 9977  045660  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
 9978  045666  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9979  045672  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
 9980  045674  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9981  045700  000240                          NOP
 9982  045702                                  5$:
 9983
 9984                                          ;REPORT ANY IVC ERROR AS
 9985                                          ;       IVC ERROR WITH VOLUME VALID ZERO
 9986                                          ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
 9987  045702  032737  010000  001372          BIT     #IVC,RMER2I     ;IVC ERROR??
 9988  045710  001427                          BEQ     52$             ;NO!!
 9989  045712  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
 9990  045716  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
 9991  045724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9992  045730  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
 9993  045736  032737  000100  001342          BIT     #VV,RMDSI
 9994  045744  001403                          BEQ     51$
 9995  045746  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
 9996  045754  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9997  045760  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9998  045762  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
 9999  045766  000240                          NOP
10000
10001  045770  013737  001372  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10002  045776  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
10003  046004  013737  046656  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
10004  046012  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
10005  046020  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
10006
10007                                          ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
10008  046022  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
10009  046030  001032                          BNE     54$             ;YES !!
10010  046032  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10011  046036  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
10012  046044  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10013  046050  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10014  046052  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10015  046056  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
10016  046060                                  53$:
```

```
10017
10018                                          ;REPORT ERROR IF SKI IS SET
10019  046060  032737  040000  001142          BIT     #SKI,$BDDAT     ;IS SKI SET ??
10020  046066  001413                          BEQ     54$             ;NO - SKI IS OK
10021  046070  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10022  046074  112776  000054  000000          MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
10023  046102  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10024  046106  004736                          JSR     PC,@(SP)+       ;REPORT SEEK ERROR
10025  046110  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10026  046114  000240                          NOP
10027
10028                                          ;REPORT ANY DEVICE CHECK
10029  046116  032737  000200  001372  54$:    BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
10030  046124  001420                          BEQ     6$              ;NO!!
10031  046126  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
10032  046132  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10033  046140  062716  000004                  ADD     #4,(SP)
10034  046144  112776  000055  000000          MOVB    #55,@(SP)       ;ERROR  #55
10035  046152  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10036  046156  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10037  046160  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10038  046164  000240                          NOP
10039
10040                                          ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
10041                                          ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
10042  046166  032737  020000  001344  6$:     BIT     #OPI,RMER1I     ;"OPI" ERROR??
10043  046174  001427                          BEQ     8$              ;NO!!
10044  046176  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
10045  046202  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10046  046210  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10047  046214  112776  000052  000000          MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
10048  046222  032737  010000  001342          BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
10049  046230  001403                          BEQ     7$              ;NO!!
10050  046232  112776  000053  000000          MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
10051  046240  162716  000002          7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10052  046244  004736                          JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
10053  046246  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10054  046252  000240                          NOP
10055
10056                                          ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
10057  046254  013746  001342          8$:     MOV     RMDSI,-(SP)
10058  046260  042716  047677                  BIC     #^C<ATA!PIP!MOL!VV>,(SP)
10059  046264  022726  110100                  CMP     #ATA!MOL!VV,(SP)+
10060  046270  001002                          BNE     9$              ;ERROR IN RMDS
10061  046272  000137  046626                  JMP     14$             ;RMDS IS OK
10062
10063                                          ;REPORT ERROR IF MOL = 0 AND OPI = 0
10064  046276  032737  010000  001342  9$:     BIT     #MOL,RMDSI      ;IS MOL RESET??
10065  046304  001030                          BNE     10$             ;NO - MOL IS SET
10066  046306  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI SET
10067  046314  001024                          BNE     10$             ;YES - DONT REPORT ERROR
10068  046316  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10069  046324  052737  010000  001140          BIS     #MOL,$GDDAT
10070  046332  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10071  046340  062716  000004                  ADD     #4,(SP)
10072  046344  112776  000062  000000          MOVB    #62,@(SP)
```

```
10073  046352  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10074  046356  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10075  046360  162716  000010            SUB     #10,(SP)
10076  046364  000240                    NOP
10077
10078                          ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
10079  046366  032737  020000  001342 10$:  BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
10080  046374  001430                    BEQ     11$              ;NO!!
10081  046376  032737  040000  001372     BIT     #SKI,RMER2I      ;WAS "SKI"SET??
10082  046404  001024                    BNE     11$              ;YES-DONT REPORT PIP
10083  046406  013737  001342  001140     MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10084  046414  042737  020000  001142     BIC     #PIP,$BDDAT
10085  046422  013737  001342  001142     MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10086  046430  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10087  046434  112776  000056  000000     MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
10088  046442  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10089  046446  004736                    JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
10090  046450  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
10091  046454  000240                    NOP
10092
10093                          ;REPORT AN ERROR IF "ATA" IS NOT SET
10094  046456  032737  100000  001342 11$:  BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
10095  046464  001024                    BNE     13$              ;YES!!
10096  046466  013737  001342  001140     MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10097  046474  052737  110600  001140     BIS     #ATA!MOL!DPR!DRY,$GDDAT
10098  046502  013737  001342  001142     MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10099  046510  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10100  046514  112776  000057  000000     MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
10101  046522  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10102  046526  004736                    JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
10103                                                           ;SEEK TEST
10104  046530  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
10105  046534  000240                    NOP
10106
10107                          ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
10108  046536  032737  000100  001342 13$:  BIT     #VV,RMDSI        ;IS VV = 0 ??
10109  046544  001030                    BNE     14$              ;NO!!
10110  046546  032737  010000  001372     BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
10111  046554  001024                    BNE     14$              ;NO - IVC IS SET
10112  046556  013737  001342  001140     MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10113  046564  052737  000100  001140     BIS     #VV,$GDDAT
10114  046572  013737  001342  001142     MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10115  046600  062716  000004            ADD     #4,(SP)
10116  046604  112776  000064  000000     MOVB    #64,@(SP)        ;ERROR #64
10117  046612  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10118  046616  004736                    JSR     PC,@(SP)+
10119  046620  162716  000010            SUB     #10,(SP)
10120  046624  000240                    NOP
10121  046626                         14$:
10122
10123                          ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10124  046626  000240                    NOP
10125  046630  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
10126  046634  105776  000000            TSTB    @(SP)            ;WAS ERROR CALLED??
10127  046640  001403                    BEQ     15$              ;NO!!
10128  046642  062716  000004            ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
10129  046646  000402                    BR      16$
10130
10131  046650  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10132  046654  000207              16$:    RTS     PC              ;RETURN
10133
10134  046656  000000              300$:   .WORD   0               ;ERROR FLAGS
10135
```

```
10136                                           .SBTTL  CONTROLLER CLEAR SUBROUTINE
10137
10138                                           ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
10139                                           ;AND DRIVES, THEN SELECTS THE DRIVE.
10140                                           ;
10141                                           ;CALL:  JSR     PC,CNTCLR
10142                                           ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10143                                           ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10144                                           ;       ERROR                   SUB DEFINES ERROR NUMBER
10145                                           ;       ???
10146
10147   046660                                  CNTCLR:
10148   046660  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10149   046662  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10150   046664  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10151   046670  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10152   046674  012737  046734  000004                  MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
10153   046702  012737  000300  000006                  MOV     #PR6,ERRVEC+2
10154   046710  013700  001276                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
10155   046714  012760  000040  000010                  MOV     #CLR,MMCS2(R0)  ;CLEAR MASSBUS
10156   046722  013701  001450                          MOV     TSTQUE,R1
10157   046726  111160  000010                          MOVB    (R1),MMCS2(R0)  ;SELECT DEVICE
10158   046732  000412                                  BR      20$
10159   046734  022626                          10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
10160   046736  062766  000004  000010                  ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
10161   046744  112776  000007  000010                  MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
10162   046752  162766  000002  000010                  SUB     #2,10(SP)
10163   046760                                  20$:
10164   046760  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10165   046764  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10166   046770  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
10167   046772  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
10168   046774  000207                                  RTS     PC
10169
```

```
10170                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
10171
10172                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
10173                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
10174                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
10175                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
10176
10177                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
10178                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10179                                   ;
10180                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
10181                                   ;
10182
10183                                   ;CALL:
10184                                   ;(1)    JSR     PC,CLRSTS
10185                                   ;       BR      ???             RETURN HERE IF NO ERROR
10186                                   ;       NOP                     RETURN HERE TO REPOR~ AN ERROR
10187                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10188                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10189                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10190
10191   046776                          CLRSTS:
10192
10193                                   ;CLEAR USER'S ERROR CALL
10194   046776  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR
10195   047002  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
10196   047006  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
10197                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10198   047012  013737  001330  001142 4$:     MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
10199   047020  042737  100000  001142         BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
10200   047026  012737  004200  001140         MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
10201   047034  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
10202   047042  001413                         BEQ     5$              ;BRANCH IF EQUAL
10203   047044  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10204   047050  112776  000126  000000         MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
10205   047056  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10206   047062  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10207   047064  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10208   047070  000240                         NOP
10209                                   ;REPORT ERROR IF RMBA NOT RESET
10210   047072  005037  001140         5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
10211   047076  013737  001334  001142         MOV     RMBAI,$BDDAT
10212   047104  001413                         BEQ     7$              ;BRANCH IF ZERO
10213   047106  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10214   047112  112776  000127  000000         MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
10215   047120  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10216   047124  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10217   047126  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10218   047132  000240                         NOP
10219                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
10220   047134  013737  001340  001142 7$:     MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
10221   047142  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10222   047144  005046                         CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
10223   047146  013701  001450                 MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
10224   047152  111116                         MOVB    (R1),(SP)
10225   047154  052716  000100                 BIS     #IR,(SP)
```

```
10226  047160  012637  001140                 MOV     (SP)+,$GDDAT
10227  047164  012601                         MOV     (SP)+,R1            ;;POP STACK INTO R1
10228  047166  023737  001140  001142         CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
10229  047174  001413                         BEQ     9$                  ;BRANCH IF EQUAL
10230  047176  062716  000004                 ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10231  047202  112776  000130  000000         MOVB    #130,@(SP)          ;WITE ERROR NUMBER IN CALL
10232  047210  162716  000002                 SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10233  047214  004736                         JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10234  047216  162716  000010                 SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10235  047222  000240                         NOP
10236                                 ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
10237  047224  005037  001140         9$:     CLR     $GDDAT              ;VERIFY RMER1
10238  047230  013737  001344  001142         MOV     RMER1I,$BDDAT
10239  047236  042737  040000  001142         BIC     #UNS,$BDDAT         ;IGNORE UNSAFE
10240  047244  001413                         BEQ     13$                 ;BRANCH IF ZERO
10241  047246  062716  000004                 ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10242  047252  112776  000131  000000         MOVB    #131,@(SP)          ;WITE ERROR NUMBER IN CALL
10243  047260  162716  000002                 SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10244  047264  004736                         JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10245  047266  162716  000010                 SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10246  047272  000240                         NOP
10247                                 ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
10248  047274  013737  001354  001142 13$:     MOV     RMMR1I,$BDDAT       ;VERIFY RMMR
10249  047302  042737  000046  001142         BIC     #WC!LS!LST,$BDDAT           ;IGNORE WORD CLOCK, SCT, TRK
10250  047310  012737  000010  001140         MOV     #WG0,$GDDAT         ;EXPECT WRITE DATA BIT
10251  047316  023737  001140  001142         CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED AND RECEIVED
10252  047324  001413                         BEQ     17$                 ;BRANCH IF 0
10253  047326  062716  000004                 ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10254  047332  112776  000133  000000         MOVB    #133,@(SP)          ;WITE ERROR NUMBER IN CALL
10255  047340  162716  000002                 SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10256  047344  004736                         JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10257  047346  162716  000010                 SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10258  047352  000240                         NOP
10259                                 ;REPORT AN ERROR IF RMEC2 IS NOT RESET
10260  047354  005037  001140         17$:    CLR     $GDDAT              ;EXPECT ZEROS
10261  047360  013737  001376  001142         MOV     RMEC2I,$BDDAT       ;VERIFY RMEC2=0
10262  047366  001413                         BEQ     19$
10263  047370  062716  000004                 ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10264  047374  112776  000135  000000         MOVB    #135,@(SP)          ;WITE ERROR NUMBER IN CALL
10265  047402  162716  000002                 SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10266  047406  004736                         JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10267  047410  162716  000010                 SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10268  047414  000240                         NOP
10269                                 ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
10270  047416  013737  001370  001142 19$:     MOV     RMMR2I,$BDDAT       ;VERIFY RMMR2
10271  047424  042737  140000  001142         BIC     #RQA!RQB,$BDDAT
10272  047432  012737  011777  001140         MOV     #TST!1777,$GDDAT            ;EXPECT TEST BIT ON
10273  047440  023737  001140  001142         CMP     $GDDAT,$BDDAT
10274  047446  001413                         BEQ     21$
10275  047450  062716  000004                 ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10276  047454  112776  000136  000000         MOVB    #136,@(SP)          ;WITE ERROR NUMBER IN CALL
10277  047462  162716  000002                 SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10278  047466  004736                         JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10279  047470  162716  000010                 SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10280  047474  000240                         NOP
10281                                 ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
10282  047476  005037  001140          21$:    CLR     $GDDAT          ;EXPECT ALL ZEROS
10283  047502  013737  001372  001142          MOV     RMER2I,$BDDAT   ;VERIFY RMER2
10284  047510  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
10285  047516  001413                          BEQ     21S$            ;BRANCH IF OTHER BITS 0
10286  047520  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10287  047524  112776  000174  000000          MOVB    #174,2(SP)      ;WITE ERROR NUMBER IN CALL
10288  047532  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10289  047536  004736                          JSR     PC,2(SP)+       ;REPORT ERROR VIA USER
10290  047540  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10291  047544  000240                          NOP
10292                                  ;REPORT ERROR IF RMOS NOT INITIALIZED
10293  047546  013737  001342  001142  21S$:    MOV     RMOSI,$BDDAT    ;TEST DRIVE STATUS REGISTER
10294  047554  042737  177177  001142          BIC     #^C<DRY!DPR>,$BDDAT
10295  047562  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
10296  047570  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10297  047576  001413                          BEQ     22$             ;BRANCH IF EQUAL
10298  047600  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10299  047604  112776  000134  000000          MOVB    #134,2(SP)      ;WRITE ERROR NUMBER
10300  047612  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10301  047616  004736                          JSR     PC,2(SP)+       ;REPORT ERROR TO USER
10302  047620  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10303  047624  000240                          NOP
10304  047626  062716  000004          22$:    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10305  047632  105776  000000                  TSTB    2(SP)           ;WAS AN ERROE DETECTED??
10306  047636  001403                          BEQ     23$             ;NO!!
10307  047640  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
10308  047644  000402                          BR      24$
10309  047646  162716  000004          23$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
10310  047652  000240                  24$:    NOP
10311  047654  000207                          RTS     PC
```

```
10312                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
10313
10314                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
10315                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
10316                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
10317
10318                                   ;CALL:
10319                                   ;(1)    JSR     PC,ACKSTS
10320                                   ;       BR      ???             RETURN HERE IF NO ERROR
10321                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10322                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10323                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10324                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10325
10326  047656                           ACKSTS:
10327
10328                                   ;CLEAR USER'S ERROR CALL
10329  047656  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10330  047662  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
10331  047666  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
10332
10333                                   ;REPORT AN ERROR  IF "VV" IS 0
10334  047672  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
10335  047700  001024                          BNE     1$              ;YES!!
10336  047702  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10337  047710  052737  000100  001140          BIS     #VV,$GDDAT
10338  047716  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10339  047724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10340  047730  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
10341  047736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10342  047742  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
10343  047744  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10344  047750  000240                          NOP
10345  047752                           1$:
10346
10347                                   ;REPORT AN ERROR IF "MOL" IS 0
10348  047752  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MOL SET??
10349  047760  001024                          BNE     2$              ;YES!!
10350  047762  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10351  047770  052737  010000  001140          BIS     #MOL,$GDDAT
10352  047776  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10353  050004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10354  050010  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10355  050016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10356  050022  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
10357  050024  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
10358  050030  000240                          NOP
10359  050032                           2$:
10360
10361                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
10362  050032  032737  060007  001344          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
10363  050040  001570                          BEQ     7$
10364
10365                                   ;REPORT AN ERROR IF "UNS" IS SET
10366  050042  032737  040000  001344          BIT     #UNS,RMER1I     ;WAS UNS SET??
10367  050050  001424                          BEQ     3$              ;NO!!
```

```
10368  050052  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10369  050060  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10370  050066  042737  040000  001140          BIC     #UNS,$GDDAT
10371  050074  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10372  050100  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10373  050106  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10374  050112  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10375  050114  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10376  050120  000240                          NOP
10377  050122                          3$:
10378
10379                                  ;REPORT ANY OPI ERROR
10380  050122  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI SET ??
10381  050130  001424                          BEQ     4$              ;NO!!
10382  050132  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10383  050140  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10384  050146  042737  020000  001140          BIC     #OPI,$GDDAT
10385  050154  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10386  050160  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10387  050166  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10388  050172  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10389  050174  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10390  050200  000240                          NOP
10391  050202                          4$:
10392
10393                                  ;REPORT ANY RMR ERROR
10394  050202  032737  000004  001344          BIT     #RMR,RMER1I     ;WAS RMR SET??
10395  050210  001424                          BEQ     5$              ;NO!!
10396  050212  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10397  050220  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10398  050226  042737  000004  001140          BIC     #RMR,$GDDAT
10399  050234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10400  050240  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10401  050246  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10402  050252  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10403  050254  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO I.O ERROR RETURN
10404  050260  000240                          NOP
10405  050262                          5$:
10406
10407                                  ;REPORT ANY     ILR ERROR
10408  050262  032737  000002  001344          BIT     #ILR,RMER1I     ;WAS ILR SET??
10409  050270  001424                          BEQ     6$              ;NO!!
10410  050272  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10411  050300  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10412  050306  042737  000002  001140          BIC     #ILR,$GDDAT
10413  050314  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10414  050320  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10415  050326  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10416  050332  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10417  050334  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10418  050340  000240                          NOP
10419  050342                          6$:
10420
10421                                  ;REPORT ANY     ILF ERROR
10422  050342  032737  000001  001344          BIT     #ILF,RMER1I     ;WAS ILF SET??
10423  050350  001424                          BEQ     7$              ;NO!!
```

```
10424  050352  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10425  050360  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10426  050366  042737  000001  001140          BIC     #1LF,$GDDAT
10427  050374  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10428  050400  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10429  050406  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10430  050412  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10431  050414  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10432  050420  000240                          NOP
10433  050422                          7$:
10434
10435                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
10436  050422  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10437  050426  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
10438  050432  001403                          BEQ     8$              ;NO!!
10439  050434  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
10440  050440  000402                          BR      9$
10441  050442  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
10442  050446  000240                  9$:     NOP
10443  050450  000207                          RTS     PC
10444
```

```
                                        .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE

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

                                        ;CALL:
                                        ;
                                        ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
                                        ;       BR      ???             ;RETURN HERE IF NO ERROR
                                        ;       NOP                     ;RETURN HERE TO REPORT AN ERROR
                                        ;       ERROR                   ;ERROR NUMBER DEFINED BY SUB
                                        ;       JSR     PC,@(SP)+       ;GO BACK TO SUB FOR MORE ERROR CHECKS
                                        ;       ???                     ;RETURN HERE IF NO MORE ERRORS
                                        ;
050452                                  RCLSTS:

                                        ;CLEAR USER'S ERROR NUMBER
050452  062716  000004                  ADD     #4,(SP)
050456  105076  000000                  CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
050462  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH

                                        ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
050466  032737  022011  001344          BIT     #OPI!PAR!ILF!IAE,RMER1I
050474  001553                          BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK

                                        ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
                                        ;"PAR" = 1 AND "DPE" = 0
050476  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS "PAR" SET??
050504  001430                          BEQ     1$              ;NO!!
050506  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS "DPE" SET??
050514  001024                          BNE     1$              ;YES - NOT A REGISTER ERROR
050516  013737  001344  001140          MOV RMER1I,$GDDAT       ;EXPECTED STATUS
050524  042737  000010  001140          BIC     #PAR,$GDDAT
050532  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
050540  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
050544  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
050552  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
050556  004736                          JSR     PC,@(SP)+       ;GO REPORT ERROR
050560  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
050564  000240                          NOP
050566                          1$:

                                        ;REPORT ANY "ILF" ERROR
050566  032737  000001  001344          BIT     #ILF,RMER1I     ;WAS "ILF" SET??
050574  001424                          BEQ     2$              ;NO!!
050576  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
050604  042737  000001  001140          BIC     #ILF,$GDDAT
050612  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
050620  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
050624  112776  000071  000000          MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
050632  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
050636  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
050640  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
050644  000240                          NOP
050646                          2$:
```

```
10501
10502                                          ;REPORT ANY "OPI" ERROR AS
10503                                          ;       . OPI DUE TO "MOL" = 0
10504                                          ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
10505  050646  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI SET??
10506  050654  001433                          BEQ     31$             ;NO!!
10507  050656  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10508  050664  042737  020000  001140          BIC     #OPI,$GDDAT
10509  050672  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10510  050700  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10511  050704  112776  000072  000000          MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10512  050712  032737  010000  001342          BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
10513  050720  001403                          BEQ     3$              ;YES!!
10514  050722  112776  000073  000000          MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
10515  050730  162716  000002          3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10516  050734  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10517  050736  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10518  050742  000240                          NOP
10519  050744                          31$:
10520
10521                                          ;REPORT AN ERROR IF "IAE" IS SET
10522  050744  032737  002000  001344          BIT     #IAE,RMER1I     ;IS "IAE" SET??
10523  050752  001424                          BEQ     4$              ;NO!!
10524  050754  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10525  050762  042737  002000  001140          BIC     #IAE,$GDDAT
10526  050770  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10527  050776  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10528  051002  112776  000070  000000          MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10529  051010  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10530  051014  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10531  051016  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
10532  051022  000240                          NOP
10533  051024                          4$:
10534
10535                                          ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
10536  051024  032737  050200  001372          BIT     #SKI!IVC!DVC,RMER2I
10537  051032  001517                          BEQ     8$              ;NONE OF THE BITS ARE SET
10538
10539
10540                                          ;REPORT ANY "IVC" ERROR AS
10541                                          ;       .IVC WITH VV = 0
10542                                          ;       .ERRONEOUS IVC ERROR
10543  051034  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS IVC SET??
10544  051042  001433                          BEQ     6$              ;NO!!
10545  051044  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10546  051052  042737  010000  001140          BIC     #IVC,$GDDAT
10547  051060  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10548  051066  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10549  051072  112776  000074  000000          MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
10550  051100  032737  000100  001342          BIT     #VV,RMDSI       ;WAS VV = 0??
10551  051106  001403                          BEQ     5$              ;YES!!
10552  051110  112776  000075  000000          MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
10553  051116  162716  000002          5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10554  051122  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10555  051124  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10556  051130  000240                          NOP
```

```
10557  051132                                  6$:
10558
10559                                          ;REPORT ANY "SKI" ERROR
10560  051132  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS SKI SET??
10561  051140  001424                          BEQ     7$              ;NO!!
10562  051142  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10563  051150  042737  040000  001140          BIC     #SKI,$GDDAT
10564  051156  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10565  051164  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10566  051170  112776  000076  000000          MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
10567  051176  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10568  051202  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10569  051204  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
10570  051210  000240                          NOP
10571  051212                                  7$:
10572
10573                                          ;REPORT ANY "DVC" ERROR
10574  051212  032737  000200  001372          BIT     #DVC,RMER2I     ;WAS "DVC" SET??
10575  051220  001424                          BEQ     8$              ;NO!!
10576  051222  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10577  051230  042737  000200  001140          BIC     #DVC,$GDDAT
10578  051236  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10579  051244  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10580  051250  112776  000077  000000          MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
10581  051256  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10582  051262  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10583  051264  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10584  051270  000240                          NOP
10585  051272                                  8$:
10586
10587                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
10588  051272  013746  001342                  MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
10589  051276  042716  047676                  BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
10590  051302  022726  110100                  CMP     #ATA!MOL!VV,(SP)+
10591  051306  001002                          BNE     85$
10592  051310  000137  051724                  JMP     13$
10593  051314                                  85$:
10594
10595                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
10596                                          ;LINE AFTER RECALIBRATE WAS INITIATED
10597  051314  032737  010000  001342          BIT     #MOL,RMDSI      ;DID MOL DROP??
10598  051322  001030                          BNE     9$              ;NO!!
10599  051324  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
10600  051332  001024                          BNE     9$              ;YES - DON'T REPORT MOL=0
10601  051334  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10602  051342  052737  010000  001140          BIS     #MOL,$GDDAT
10603  051350  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10604  051356  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10605  051362  112776  000100  000000          MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
10606  051370  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10607  051374  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10608  051376  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10609  051402  000240                          NOP
10610  051404                                  9$:
10611
10612                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
10613  051404  032737  000100  001342          BIT     #VV,RMDSI       ;DID "VV" DROP??
10614  051412  001030                          BNE     10$             ;NO!!
10615  051414  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
10616  051422  001024                          BNE     10$             ;YES - DONT REPORT VV=0
10617  051424  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10618  051432  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10619  051440  052737  000100  001140          BIS     #VV,$GDDAT
10620  051446  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10621  051452  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
10622  051460  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10623  051464  004736                          JSR     PC,@(SP)+
10624  051466  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10625  051472  000240                          NOP
10626  051474                          10$:
10627
10628                                  ;REPORT AN ERROR IF ATA IS NOT SET
10629  051474  032737  100000  001342          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
10630  051502  001024                          BNE     11$             ;YES!!
10631  051504  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10632  051512  052737  100000  001140          BIS     #ATA,$GDDAT
10633  051520  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10634  051526  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10635  051532  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
10636  051540  162716  000002                  SUB     #2,(SP)
10637  051544  004736                          JSR     PC,@(SP)+
10638  051546  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10639  051552  000240                          NOP
10640
10641  051554                          11$:
10642
10643                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
10644                                  ;ALWAYS CLEAR OFFSET MODE
10645  051554  032737  000001  001342          BIT     #OM,RMDSI       ;WAS "OM" RESET??
10646  051562  001424                          BEQ     12$             ;YES!!
10647  051564  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10648  051572  042737  000001  001140          BIC     #OM,$GDDAT
10649  051600  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10650  051606  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10651  051612  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
10652  051620  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10653  051624  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10654  051626  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10655  051632  000240                          NOP
10656  051634                          12$:
10657
10658                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
10659                                  ;CYLINDER
10660  051634  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
10661  051642  001430                          BEQ     13$             ;NO!!
10662  051644  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
10663  051652  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
10664  051654  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10665  051662  042737  020000  001140          BIC     #PIP,$GDDAT
10666  051670  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10667  051676  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10668  051702  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
10669  051710  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10670  051714  004736                  JSR     PC,@(SP)+
10671  051716  162716  000010          SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
10672  051722  000240                  NOP
10673  051724                  13$:
10674
10675                          ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
10676  051724  032737  040006  001344          BIT     #ILR!RMR!UNS,RMER1I
10677  051732  001514                  BEQ     16$
10678
10679                          ;REPORT AN ERROR IF "ILR" IS SET
10680  051734  032737  000002  001344          BIT     #ILR,RMER1I      ;WAS ILR SET DURING RECALIBRATE??
10681  051742  001424                  BEQ     14$              ;NO!!
10682  051744  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10683  051752  042737  000002  001140          BIC     #ILR,$GDDAT
10684  051760  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10685  051766  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10686  051772  112776  000105  000000          MOVB    #105,@(SP)       ;WRITE ERROR NUMBER IN CALL
10687  052000  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10688  052004  004736                  JSR     PC,@(SP)+
10689  052006  162716  000010          SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
10690  052012  000240                  NOP
10691  052014                  14$:
10692
10693                          ;REPORT AN ERROR IF "RMR" IS SET
10694  052014  032737  000004  001344          BIT     #RMR,RMER1I      ;WAS RMR SET??
10695  052022  001424                  BEQ     15$              ;NO!!
10696  052024  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10697  052032  042737  000004  001140          BIC     #RMR,$GDDAT
10698  052040  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10699  052046  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
10700  052052  112776  000106  000000          MOVB    #106,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10701  052060  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10702  052064  004736                  JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10703  052066  162716  000010          SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
10704  052072  000240                  NOP
10705  052074                  15$:
10706
10707                          ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
10708  052074  032737  040000  001344          BIT     #UNS,RMER1I      ;WAS UNSAFE ON??
10709  052102  001430                  BEQ     16$              ;NO!!
10710  052104  032737  000200  001372          BIT     #DVC,RMER2I      ;WAS THERE A DEVICE CHECK??
10711  052112  001024                  BNE     16$              ;YES - DON'T REPORT UNSAFE
10712  052114  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10713  052122  042737  040000  001140          BIC     #UNS,$GDDAT
10714  052130  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10715  052136  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
10716  052142  112776  000107  000000          MOVB    #107,@(SP)       ;WRITE ERROR NUMBER
10717  052150  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10718  052154  004736                  JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10719  052156  162716  000010          SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
10720  052162  000240                  NOP
10721  052164                  16$:
10722
10723                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
10724  052164  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
```

```
10725  052170  105776  000000                  TSTB    @(SP)             ;WAS AN ERROR REPORTED??
10726  052174  001403                          BEQ     17$               ;NO!!
10727  052176  062716  000004                  ADD     #4,(SP)           ;YES - AUGMENT SP RETURN
10730  052202  000402                          BR      18$
10731  052204  162716  000004          17$:    SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
10732  052210  000240          18$:    NOP
10733  052212  000207                          RTS     PC                ;STATUS CECK IS COMPLETE
10734
```

```
10733                                           .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
10734                                           ;       BR      ???             RETURN HERE IF NO ERROR
10735                                           ;       NOP                     RETURN HERE TO REPORT AN ERROR
10736                                           ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10737                                           ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10738                                           ;       ???                     RETURN HERE IF NO MORE ERRORS
10739
10740   052214                                  DRVSTS:
10741
10742                                           ;CLEAR USER'S ERROR CALL
10743   052214  062716  000004                         ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10744   052220  005076  000000                         CLR     @(SP)           ;CLEAR ERROR CALL
10745   052224  162716  000004                         SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
10746                                           ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10747   052230  013737  001330  001142         4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
10748   052236  042737  173700  001142                 BIC     #↑C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
10749   052244  012737  004010  001140                 MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
10750   052252  023737  001140  001142                 CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10751   052260  001443                                 BEQ     6$              ;BRANCH IF EQUAL
10752   052262  062716  000004                         ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10753   052266  112776  000141  000000                 MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10754   052274  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10755   052300  004736                                 JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10756   052302  162716  000010                         SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10757   052306  000240                                 NOP
10758                                           ;REPORT ERROR IF RMDS NOT INITIALIZED
10759   052310  013737  001342  001142         5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
10760   052316  042737  021101  001142                 BIC     #PGM!OM!VV,PIP,$BDDAT   ;CLEAR DONT CARES
10761   052324  012737  010600  001140                 MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
10762   052332  023737  001140  001142                 CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10763   052340  001413                                 BEQ     6$              ;BRANCH IF EQUAL
10764   052342  062716  000004                         ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10765   052346  112776  000142  000000                 MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10766   052354  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10767   052360  004736                                 JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10768   052362  162716  000010                         SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10769   052366  000240                                 NOP
10770                                           ;REPORT ERROR IF RMER1 NOT INITIALIZED
10771   052370  005037  001140                 6$:     CLR     $GDDAT          ;EXPECT O'S
10772   052374  013737  001344  001142                 MOV     RMER1I,$BDDAT   ;CHECK RMER1
10773   052402  001413                                 BEQ     8$              ;BRANCH IF EQUAL
10774   052404  062716  000004                         ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10775   052410  112776  000143  000000                 MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10776   052416  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10777   052422  004736                                 JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10778   052424  162716  000010                         SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10779   052430  000240                                 NOP
10780                                           ;REPORT ERROR IF ATA NOT INITIALIZED
10781   052432  013737  001346  001142         8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
10782   052440  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10783   052442  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10784   052444  013701  001450                         MOV     TSTQUE,R1
10785   052450  116102  000001                         MOVB    1(R1),R2
10786   052454  042702  177400                         BIC     #↑CATNMSK,R2
10787   052460  005102                                 COM     R2
10788   052462  040237  001142                         BIC     R2,$BDDAT
```

```
10789  052466  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10790  052470  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10791  052472  005737  001142                  TST     $BDDAT          ;IS ATTENTION CLEARED??
10792  052476  001413                          BEQ     9$              ;BRANCH IF ATTENTION CLEARED
10793  052500  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10794  052504  112776  000144  000000          MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10795  052512  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10796  052516  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10797  052520  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10798  052524  000240                          NOP
10799                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
10800  052526  013737  001354  001142  9$:     MOV     RMMR1I,$BDDAT   ;CHECK RMMR
10801  052534  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
10802  052542  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
10803  052550  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10804  052556  001413                          BEQ     11$             ;BRANCH IF ZERO
10805  052560  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10806  052564  112776  000145  000000          MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10807  052572  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10808  052576  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10809  052600  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10810  052604  000240                          NOP
10811                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
10812  052606  013737  001370  001142  11$:     MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
10813  052614  042737  140000  001142          BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
10814  052622  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10815  052630  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10816  052636  001413                          BEQ     15$             ;BRANCH IF EQUAL
10817  052640  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10818  052644  112776  000146  000000          MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10819  052652  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10820  052656  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10821  052660  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10822  052664  000240                          NOP
10823  052666  005037  001140          15$:    CLR     $GDDAT          ;EXPECT ZEROS
10824                          ;REPORT ERROR IF RMEC2 NOT RESET
10825  052672  013737  001376  001142          MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
10826  052700  001413                          BEQ     17$             ;BRANCH IF 0
10827  052702  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10828  052706  112776  000150  000000          MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10829  052714  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10830  052720  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10831  052722  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10832  052726  000240                          NOP
10833                          ;REPORT ERROR IF RMER2 NOT RESET
10834  052730  013737  001372  001142  17$:     MOV     RMER2I,$BDDAT   ;CHECK RMER2
10835  052736  001413                          BEQ     18$             ;BRANCH IF NO ERROR
10836  052740  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10837  052744  112776  000147  000000          MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10838  052752  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10839  052756  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10840  052760  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10841  052764  000240                          NOP
10842  052766                          18$:
10843
10844  052766                          19$:
```

```
10845
10846                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
10847   052766  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10848   052772  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
10849   052776  001403                          BEQ     21$             ;NO!!
10850   053000  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
10851   053004  000402                          BR      23$
10852   053006  162716  000004          21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
10853   053012  000240                  23$:    NOP
10854   053014  000207                          RTS     PC              ;RETURN TO USER
```

```
                                        .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

                                        ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
                                        ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
                                        ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
                                        ;THE ERROR NUMBER IN THE USERS ERROR CALL.

                                        ;USER'S SUBROUTINE CALL:
                                        ;(1)    JSR     PC,DTASTS
                                        ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
                                        ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
                                        ;(4)    ERROR                   SUB WRITES ERROR NUMBER
                                        ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
                                        ;(6)    ??                      SUB RETURNS HERE AFTER ALL
                                        ;                               ERRORS ARE REPORTED

053016                                  DTASTS:

                                        ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
                                                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
053016  062716  000004                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
053022  105076  000000                          SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
053026  162716  000004                          CLR     500$            ;CLEAR ERROR FLAGS
053032  005037  056412

                                        ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
053036  032737  020000  001330                  BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
053044  001422                                  BEQ     10$             ;NO!!
053046  013737  001330  001140                  MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
053054  042737  020000  001140                  BIC     #MCPE,$GDDAT
053062  013737  001330  001142                  MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
053070  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
053074  112775  000013  000000                  MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
053102  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
053106  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
053110  000466                                  BR      30$
053112                                  10$:

                                        ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
                                        ;PAR=1  AND DPE = 0
053112  032737  000010  001344                  BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
053120  001435                                  BEQ     20$             ;NO!!
053122  032737  000010  001372                  BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
053130  001031                                  BNE     20$             ;NO!!
053132  013737  001344  001140                  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
053140  042737  000010  001140                  BIC     #PAR,$GDDAT
053146  013737  001344  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
053154  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
053160  112776  000050  000000                  MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
053166  032737  001000  001330                  BIT     #MXF,RMCS1I     ;DID MXF GET SET??
053174  001003                                  BNE     15$             ;YES!!
053176  112776  000274  000000                  MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
053204  162716  000002                  15$:    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
053210  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
053212  000425                                  BR      30$

053214                                  20$:
```

```
10911
10912                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
10913                                   ;MECHANICAL POSITIONING
10914
10915                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
10916                                   ;CODE AND GO BIT WERE LOADED
10917  053214  032737  001000  001340           BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
10918  053222  001425                           BEQ     40$             ;NO!!
10919  053224  013737  001340  001140           MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
10920  053232  042737  001000  001140           BIC     #MXF,$GDDAT
10921  053240  013737  001340  001142           MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10922  053246  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10923  053252  112776  000275  000000           MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
10924  053260  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10925  053264  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10926  053266                          30$:
10927
10928                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
10929  053266  162716  000010                   SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10930  053272  000137  056364                   JMP     380$            ;SKIP TO END OF SUB
10931
10932  053276                          40$:
10933
10934                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
10935                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
10936                                   ;"MOL"
10937  053276  032737  020000  001344           BIT     #OPI,RMER1I     ;IS "OPI" SET??
10938  053304  001447                           BEQ     60$             ;NO!!
10939  053306  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10940  053314  042737  020000  001140           BIC     #OPI,$GDDAT
10941  053322  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10942  053330  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
10943  053336  001404                           BEQ     45$             ;YES!!
10944  053340  032737  000100  001342           BIT     #VV,RMDSI       ;WAS "MOL" INTERMITTENT??
10945  053346  001013                           BNE     50$             ;NO!!
10946  053350  062716  000004          45$:     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10947  053354  112776  000276  000000           MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
10948  053362  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10949  053366  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10950  053370  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10951  053374  000413                           BR      60$
10952  053376                          50$:
10953
10954                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
10955                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
10956  053376  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10957  053402  112776  000277  000000           MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
10958  053410  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10959  053414  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10960  053416  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10961  053422  000240                           NOP
10962  053424                          60$:
10963
10964                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
10965  053424  032737  010000  001372           BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
10966  053432  001432                           BEQ     70$             ;NO!!
```

```
10967                                           ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
10968  053434  013737  001372  001140                   MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10969  053442  042737  010000  001140                   BIC     #IVC,$GDDAT
10970  053450  013737  001372  001142                   MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10971  053456  062716  000004                           ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10972  053462  112776  000300  000000                   MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
10973  053470  032737  000100  001342                   BIT     #VV,RMDSI       ;WAS VOLUME VALID??
10974  053476  001403                                   BEQ     65$             ;NO!!
10975  053500  112776  000301  000000                   MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
10976  053506  162716  000002                 65$:      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10977  053512  004736                                   JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
10978  053514  162716  000010                           SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10979  053520                                 70$:
10980
10981                                           ;SEE IF "ILF" OR "RMR" IS SET
10982  053520  032737  000007  001344                   BIT     #ILR!ILF!RMR, RMER1I
10983  053526  001510                                   BEQ     100$            ;NO ERRORS DETECTED
10984                                           ;REPORT AN ERROR IF "ILR" IS SET
10985  053530  032737  000002  001344                   BIT     #ILR,RMER1I     ;WAS "ILR" DETECTED??
10986  053536  001424                                   BEQ     80$             ;NO!!
10987  053540  013737  001344  001140                   MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10988  053546  042737  000002  001140                   BIC     #ILR,$GDDAT
10989  053554  013737  001344  001142                   MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10990  053562  062716  000004                           ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10991  053566  112776  000302  000000                   MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
10992  053574  162716  000002                           SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10993  053600  004736                                   JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10994  053602  162716  000010                           SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10995  053606  000240                                   NOP
10996  053610                                 80$:
10997
10998                                           ;REPORT AN ERROR IF "ILF" IS SET
10999  053610  032737  000001  001344                   BIT     #ILF,RMER1I     ;WAS "ILF" DETECTED??
11000  053616  001424                                   BEQ     90$             ;NO!!
11001  053620  013737  001344  001140                   MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11002  053626  042737  000001  001140                   BIC     #ILF,$GDDAT
11003  053634  013737  001344  001142                   MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11004  053642  062716  000004                           ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11005  053646  112776  000303  000000                   MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
11006  053654  162716  000002                           SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11007  053660  004736                                   JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
11008  053662  162716  000010                           SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11009  053666  000240                                   NOP
11010  053670                                 90$:
11011
11012                                           ;REPORT AN ERROR IF "RMR" IS SET
11013  053670  032737  000004  001344                   BIT     #RMR,RMER1I     ;WAS "RMR" DETECTED??
11014  053676  001424                                   BEQ     100$            ;NO!!
11015  053700  013737  001344  001140                   MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11016  053706  042737  000004  001140                   BIC     #RMR,$GDDAT
11017  053714  013737  001344  001142                   MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11018  053722  062716  000004                           ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11019  053726  112776  000304  000000                   MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
11020  053734  162716  000002                           SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11021  053740  004736                                   JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11022  053742  162716  000010                           SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
11023  053746  000240                          NOP
11024  053750                          100$:
11025                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
11026  053750  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
11027  053756  052737  040000  056412          BIS     #SKI,500$       ;SET SKI FLAG
11028  053764  022737  001466  001434          CMP     #822.,RMDC0     ;IS CYLINDER > 822??
11029  053772  103425                          BLO     110$            ;YES!!
11030  053774  042737  040000  056412          BIC     #SKI,500$       ;RESET SKI FLAG
11031  054002  122737  000004  001407          CMPB    #4,RMDA0+1      ;IS TRACK > 4??
11032  054010  103416                          BLO     110$            ;YES!!
11033  054012  122737  000037  001406          CMPB    #31.,RMDA0      ;IS SECTOR > 31??
11034  054020  103412                          BLO     110$            ;YES!!
11035  054022  122737  000035  001406          CMPB    #29.,RMDA0      ;IS SECTOR > 29??
11036  054030  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
11037  054032  032737  010000  001432          BIT     #FMT16,RMOF0    ;18 BIT FORMAT??
11038  054040  001402                          BEQ     110$            ;YES!!
11039  054042  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
11040  054046  013737  001344  001142  110$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
11041  054054  042737  175777  001142          BIC     #^C<IAE>,$BDDAT
11042  054062  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
11043  054070  001004                          BNE     115$            ;NO!!
11044  054072  042737  040000  056412          BIC     #SKI,500$       ;IAE OK - SKI SHOULD BE 0
11045  054100  000412                          BR      120$
11046  054102  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11047  054106  112776  000305  000000          MOVB    #305,@(SP)      ;WRITE ERROR NUMBER
11048  054114  162716  000002                  SUB     #2.,(SP)        ;MOVE SP TO RETURN IF ERROR
11049  054120  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11050  054122  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11051  054126                          120$:
11052
11053                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
11054  054126  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11055  054134  042737  137777  001142          BIC     #^C<SKI>,$BDDAT
11056  054142  013737  056412  001140          MOV     500$,$GDDAT     ;EXPECTED STATUS
11057  054150  042737  137777  001140          BIC     #^C<SKI>,$GDDAT
11058  054156  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11059  054164  001417                          BEQ     140$            ;NO!!
11060  054166  032737  040000  056412          BIT     #SKI,500$       ;WAS SKI CAUSED BY IAE = 0??
11061  054174  001032                          BNE     150$            ;YES - DON'T REPORT SKI
11062  054176  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11063  054202  112776  000306  000000          MOVB    #306,@(SP)      ;WRITE ERROR NUMBER
11064  054210  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11065  054214  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11066  054216  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11067  054222  000417                          BR      150$
11068
11069  054224                          140$:
11070
11071                                  ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
11072  054224  032737  040000  056412          BIT     #SKI,500$       ;SHOULD SKI BE SET??
11073  054232  001413                          BEQ     150$            ;NO!!
11074  054234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11075  054240  112776  000307  000000          MOVB    #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
11076  054246  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11077  054252  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11078  054254  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
11079  054260  000240                          NOP
11080  054262                          150$:
11081
11082                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
11083  054262  032737  006200  001372          BIT     #LSC!LBC!DVC,RMER2I
11084  054270  001512                          BEQ     180$            ;NO ERRORS SET
11085
11086                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
11087  054272  032737  000200  001372          BIT     #DVC,RMER2I     ;IS "DVC" = 1??
11088  054300  001424                          BEQ     160$            ;NO!!
11089  054302  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11090  054310  042737  000200  001140          BIC     #DVC,$GDDAT
11091  054316  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11092  054324  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11093  054330  112776  000310  000000          MOVB    #310,@(SP)      ;WRITE ERROR NUMBER IN CALL
11094  054336  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11095  054342  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11096  054344  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11097  054350  000240                          NOP
11098  054352                          160$:
11099
11100                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
11101  054352  032737  002000  001372          BIT     #LBC,RMER2I     ;IS LBC SET??
11102  054360  001430                          BEQ     170$            ;NO!
11103  054362  032737  010000  001342          BIT     #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
11104  054370  001424                          BEQ     170$            ;YES!!
11105  054372  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11106  054400  042737  002000  001140          BIC     #LBC,$GDDAT
11107  054406  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11108  054414  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11109  054420  112776  000311  000000          MOVB    #311,@(SP)      ;WRITE ERROR NUMBER IN CALL
11110  054426  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11111  054432  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11112  054434  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11113  054440  000240                          NOP
11114  054442                          170$:
11115
11116                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
11117  054442  032737  004000  001372          BIT     #LSC,RMER2I     ;IS "LSC" = 1??
11118  054450  001422                          BEQ     180$            ;NO!!
11119  054452  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11120  054460  042737  004000  001140          BIC     #LSC,$GDDAT
11121  054466  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11122  054474  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11123  054500  112776  000312  000000          MOVB    #312,@(SP)      ;WRITE ERROR NUMBER
11124  054506  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11125  054510  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11126  054514  000240                          NOP
11127  054516                          180$:
11128
11129                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
11130  054516  032737  054000  001344          BIT     #UNS!DTE!WLE,RMER1I
11131  054524  001527                          BEQ     220$            ;NO BITS SET
11132                                  ;REPORT "UNS" ERROR IF "DVC" = 0
11133  054526  032737  040000  001344          BIT     #UNS,RMER1I     ;IS "UNS" SET??
11134  054534  001427                          BEQ     190$            ;NO!!
```

```
11135  054536  032737  000200  001372          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
11136  054544  001023                          BNE     190$            ;YES!!
11137  054546  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11138  054554  042737  040000  001140          BIC     #UNS,$GDDAT
11139  054562  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11140  054570  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11141  054574  112776  000313  000000          MOVB    #313,@(SP)      ;WRITE ERROR NUMBER
11142  054602  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11143  054606  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11144  054610  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11145  054614                          190$:
11146
11147                                  ;REPORT ANY DRIVE TIMING ERROR, I.E., "DTE" = 1
11148  054614  032737  010000  001344          BIT     #DTE,RMER1I     ;IS DTE SET??
11149  054622  001423                          BEQ     200$            ;NO!!
11150  054624  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11151  054632  042737  010000  001140          BIC     #DTE,$GDDAT
11152  054640  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11153  054646  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11154  054652  112776  000314  000000          MOVB    #314,@(SP)      ;WRITE ERROR NUMBER IN CALL
11155  054660  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11156  054664  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11157  054666  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11158  054672                          200$:
11159
11160                                  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
11161                                  ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
11162  054672  032737  004000  001344          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
11163  054700  001441                          BEQ     220$            ;NO!!
11164  054702  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11165  054710  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11166  054716  052737  004000  001140          BIS     #WLE,$GDDAT
11167  054724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11168  054730  112776  000315  000000          MOVB    #315,@(SP)      ;WRITE ERROR NUMBER IN CALL
11169  054736  032737  004000  001342          BIT     #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
11170  054744  001404                          BEQ     205$            ;NO!!
11171  054746  032737  000010  001400          BIT     #BIT3,RMCS1O    ;WAS COMMAND A WRITE??
11172  054754  001406                          BEQ     210$            ;YES!!
11173  054756  112776  000316  000000  205$:   MOVB    #316,@(SP)      ;CHANGE ERROR NUMBER
11174  054764  042737  004000  001140          BIC     #WLE,$GDDAT
11175  054772  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11176  054776  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11177  055000  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11178
11179  055004                          220$:
11180
11181                                  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
11182  055004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11183  055010  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
11184  055014  001404                          BEQ     225$            ;NO - DO DATA CHECKS
11185  055016  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
11186  055022  000137  056024                  JMP     340$            ;SKIP DATA CHECKS
11187  055026  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
11188
11189                                  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, ANC
11190                                  ;IF HEADER COMPARE IS NOT INHIBITED
```

```
11191  055032  013737  001400  056414          MOV     RMCS10,S10$     ;STRIP AND STORE FUNCTION CODE
11192  055040  042737  177700  056414          BIC     #↑CFNCMSK,S10$
11193  055046  022737  000063  056414          CMP     #WH!GO,S10$     ;WAS FUNCTION WRITE HEADER & DATA??
11194  055054  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
11195  055056  032737  002000  001362          BIT     #HCI,RMOFI      ;WAS HCI SET??
11196  055064  001105                          BNE     250$            ;YES - SKIP HEADER CHECKS
11197
11198                                  ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
11199  055066  032737  000620  001344          BIT     #HCRC!FER!HCE,RMER1I
11200  055074  001533                          BEQ     270$            ;NO ERRORS SET
11201
11202                                  ;REPORT HEADER CRC ERROR IF SET
11203  055076  032737  000400  001344          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
11204  055104  001422                          BEQ     230$            ;NO!!
11205  055106  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11206  055114  042737  000400  001140          BIC     #HCRC,$GDDAT
11207  055122  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11208  055130  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11209  055134  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
11210  055142  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11211  055146  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11212  055150  000501                          BR      260$
11213  055152                          230$:
11214
11215                                  ;REPORT FORMAT ERROR IF SET
11216  055152  032737  000020  001344          BIT     #FER,RMER1I     ;WAS "FER" SET??
11217  055160  001422                          BEQ     240$            ;NO!!
11218  055162  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11219  055170  042737  000020  001140          BIC     #FER,$GDDAT
11220  055176  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11221  055204  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11222  055210  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
11223  055216  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11224  055222  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11225  055224  000453                          BR      260$
11226  055226                          240$:
11227
11228                                  ;REPORT HEADER COMPARE ERROR IF SET
11229  055226  032737  000200  001344          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
11230  055234  001453                          BEQ     270$            ;NO!!
11231  055236  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11232  055244  042737  000200  001140          BIC     #HCE,$GDDAT
11233  055252  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11234  055260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11235  055264  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
11236  055272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11237  055276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11238  055300  000425                          BR      260$
11239                                  ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
11240                                  ;       .COMMAND WAS WRITE HEADER AND DATA, OR
11241                                  ;       .HEADER COMPARE INHIBIT WAS SET
11242  055302  032737  000620  001344  250$:   BIT     #HCE!FER!HCRC,RMER1I
11243  055310  001425                          BEQ     270$            ;NO ERRORS WERE SET
11244  055312  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11245  055320  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
11246  055326  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
11247  055334  062716  000004                        ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11248  055340  112776  000322  000000                MOVB    #322,@(SP)       ;WRITE ERROR NUMBER
11249  055346  162716  000002                        SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11250  055352  004736                                JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11251  055354  162716  000010              260$:     SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11252  055360  000137  056024                        JMP     340$             ;OMIT FURTHER DATA CHECKS
11253
11254  055364                              270$:
11255
11256                                      ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
11257                                      ;DO READ ERROR CHECKS
11258  055364  032737  000010  056414                BIT     #BIT3,510$       ;WAS THIS A WRITE COMMAND?
11259  055372  001002                                BNE     275$             ;NO!!
11260  055374  000137  055612                        JMP     310$             ;GO DO WRITE STATUS CHECK
11261  055400                              275$:
11262
11263                                      ;REPORT DATA CHECK IF SET
11264  055400  032737  100000  001344                BIT     #DCK,RMER1I      ;DATA CHECK ERROR??
11265  055406  001450                                BEQ     290$             ;NO!!
11266  055410  013737  001344  001140                MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11267  055416  042737  100000  001140                BIC     #DCK,$GDDAT
11268  055424  013737  001344  001142                MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11269  055432  062716  000004                        ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11270  055436  112776  000323  000000                MOVB    #323,@(SP)       ;WRITE ERROR NUMBER
11271  055444  032737  004000  001362                BIT     #ECI,RMOFI       ;WAS ECC CORRECTION DISABLED??
11272  055452  001021                                BNE     280$             ;YES!!
11273  055454  112776  000324  000000                MOVB    #324,@(SP)       ;CHANGE TO RECOVERABLE ERROR
11274  055462  032737  000100  001344                BIT     #ECH,RMER1I      ;IS ERROR RECOVERABLE??
11275  055470  001007                                BNE     276$             ;NO !!
11276                                      ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
11277  055472  032737  000020  056414                BIT     #BIT4,510$       ;WAS THIS A READ COMMAND ??
11278  055500  001406                                BEQ     280$             ;NO !!
11279  055502  162716  000004                        SUB     #4,(SP)          ;RESTORE SP
11280  055506  000410                                BR      290$             ;SKIP ERROR - DATA WILL BE CORRECTED
11281  055510  112776  000325  000000      276$:     MOVB    #325,@(SP)       ;CHANGE TO NON RECOVERABLE
11282  055516  162716  000002              280$:     SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11283  055522  004736                                JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11284  055524  162716  000010                        SUB     #10,(SP)         ;RESTORE SP TO NO ERROR
11285
11286  055530                              290$:
11287
11288                                      ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
11289  055530  032737  000400  001340                BIT     #MDPE,RMCS2I     ;PARITY ERROR SET??
11290  055536  001423                                BEQ     300$             ;NO!!
11291  055540  013737  001340  001140                MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
11292  055546  042737  000400  001140                BIC     #MDPE,$GDDAT
11293  055554  013737  001340  001142                MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
11294  055562  062716  000004                        ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11295  055566  112776  000326  000000                MOVB    #326,@(SP)       ;WRITE ERROR NUMBER
11296  055574  162716  000002                        SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11297  055600  004736                                JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11298  055602  162716  000010                        SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11299  055606  000137  056024              300$:     JMP     340$             ;SKIP WRITE STATUS CHECK
11300
11301  055612                              310$:
11302
```

```
11303                                          ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
11304  055612  032737  000001  001342                BIT     #OM,RMDSI       ;IS OFFSET ON??
11305  055620  001423                              BEQ     320$            ;NO
11306  055622  013737  001342  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11307  055630  042737  000001  001140              BIC     #OM,$GDDAT
11308  055636  013737  001342  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11309  055644  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11310  055650  112776  000327  000000              MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
11311  055656  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11312  055662  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11313  055664  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11314  055670                                 320$:
11315
11316                                          ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
11317  055670  032737  000010  001372                BIT     #DPE,RMER2I     ;DATA PARITY ERROR??
11318  055676  001423                              BEQ     330$            ;NO!!
11319  055700  013737  001372  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11320  055706  042737  000010  001140              BIC     #DPE,$GDDAT
11321  055714  013737  001372  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11322  055722  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11323  055726  112776  000330  000000              MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
11324  055734  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11325  055740  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11326  055742  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11327  055746                                 330$:
11328
11329                                          ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
11330  055746  032737  000040  001344                BIT     #WCF,RMER1I     ;IS "WCF" SET??
11331  055754  001423                              BEQ     340$            ;NO!!
11332  055756  013737  001344  001140              MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11333  055764  042737  000040  001140              BIC     #WCF,$GDDAT
11334  055772  013737  001344  001142              MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11335  056000  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11336  056004  112776  000331  000[ 3              MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
11337  056012  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11338  056016  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11339  056020  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11340  056024                                 340$:
11341
11342                                          ;REPORT "DATA LATE" ERROR IF "DLT" = 1
11343  056024  032737  100000  001340                BIT     #DLT,RMCS2I     ;IS "DLT" SET??
11344  056032  001423                              BEQ     350$            ;NO!!
11345  056034  013737  001340  001140              MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
11346  056042  042737  100000  001140              BIC     #DLT,$GDDAT
11347  056050  013737  001340  001142              MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
11348  056056  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11349  056062  112776  000332  000000              MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
11350  056070  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11351  056074  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11352  056076  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11353  056102                                 350$:
11354                                          ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
11355  056102  013746  001342                      MOV     RMDSI,-(SP)     ;STACK DRIVE STATUS
11356  056106  042716  147677                      BIC     #↑C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
11357  056112  022726  010100                      CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
11358  056116  001522                              BEQ     380$            ;YES!!
```

```
11359
11360                                   ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE EPROR,
11361                                   ;I.E, PIP = 1 AND SKI = 0
11362  056120  032737  020000  001342            BIT     #PIP,RMDSI      ;IS "PIP" SET??
11363  056126  001430                            BEQ     360$            ;NO!!
11364  056130  032737  040000  001372            BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
11365  056136  001024                            BNE     360$            ;YES-DONT REPORT PIP
11366  056140  013737  001342  001140            MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11367  056146  042737  020000  001140            BIC     #PIP,$GDDAT
11368  056154  013737  001342  001142            MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11369  056162  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11370  056166  112776  000333  000000            MOVB    #333,2(SP)      ;WRITE ERROR NUMBER
11371  056174  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11372  056200  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11373  056202  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11374  056206  000240                            NOP
11375  056210                          360$:
11376
11377                                   ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
11378                                   ;REPORTED, I.E., MOL = OPI = 0
11379  056210  032737  010000  001342            BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
11380  056216  001027                            BNE     370$            ;YES!!
11381  056220  032737  020000  001344            BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11382  056226  001023                            BNE     370$            ;YES!!
11383  056230  013737  001342  001140            MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11384  056236  052737  010000  001140            BIS     #MOL,$GDDAT
11385  056244  013737  001342  001142            MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11386  056252  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11387  056256  112776  000334  000000            MOVB    #334,2(SP)      ;WRITE ERROR NUMBER
11388  056264  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11389  056270  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11390  056272  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11391  056276                          370$:
11392
11393                                   ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
11394                                   ;REPORTED, I.E., VV = IVC = 0
11395  056276  032737  000100  001342            BIT     #VV,RMDSI       ;IS VOLUME VALID??
11396  056304  001027                            BNE     380$            ;YES!!
11397  056306  032737  010000  001372            BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
11398  056314  001033                            BNE     390$            ;YES!!
11399  056316  013737  001342  001140            MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11400  056324  052737  000100  001140            BIS     #VV,$GDDAT
11401  056332  013737  001342  001142            MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11402  056340  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11403  056344  112776  000335  000000            MOVB    #335,2(SP)      ;WRITE ERROR NUMBER
11404  056352  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11405  056356  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11406  056360  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11407  056364                          380$:
11408
11409                                   ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
11410  056364  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO EPROR CALL
11411  056370  105776  000000                    TSTB    @(SP)           ;ANY ERROR??
11412  056374  001403                            BEQ     390$            ;NO!!
11413  056376  062716  000004                    ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11414  056402  000402                            BR      400$
```

```
11415  056404  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11416
11417  056410  000207                  400$:   RTS     PC              ;RETURN TO USER
11418
11419  056412  000000                  500$:   .WORD                   ;ERROR FLAGS
11420  056414  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
 11421                                  .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
 11422
 11423                                  ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
 11424                                  ;STATUS,  SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
 11425                                  ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
 11426                                  ;SHOULD NO. BE ANY DRIVE ERRORS OR CHANGES IN STATE.
 11427
 11428                                  ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
 11429                                  ;IF TRUE:
 11430
 11431                                  ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
 11432                                  ;THAT MOL IS ASSUMED TO HAVE BEEN SET
 11433                                  ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
 11434                                  ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
 11435                                  ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
 11436                                  ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
 11437
 11438                                  ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
 11439
 11440                                  ;(1)    JSR     PC,STCDRVSTS
 11441                                  ;       BR      ???             RETURN HERE IF NO ERROR
 11442                                  ;       NOP                     RETURN HERE TO REPORT AN ERROR
 11443                                  ;       ERROR                   ERROR NUMBER DEFINED BY SUB
 11444                                  ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
 11445                                  ;       ???                     RETURN HERE IF NO MORE ERRORS
 11446
 11447  056416                          STCDRVSTS:
 11448
 11449                                  ;CLEAR USER'S ERROR CALL
 11450  056416  062716  000004                 ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 11451  056422  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
 11452  056426  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
 11453                                  ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
 11454  056432  013746  001342                 MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
 11455  056436  042716  147677                 BIC     #^C<PIP!MOL!VV>,(SP)
 11456  056442  022726  010100                 CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
 11457  056446  001524                         BEQ     30$             ;YES!!
 11458
 11459                                  ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
 11460  056450  032737  010000  001342         BIT     #MOL,RMDSI      ;IS MOL ON ??
 11461  056456  001030                         BNE     10$             ;YES!!
 11462  056460  032737  020000  001344         BIT     #OPI,RMER1I     ;WAS "OPI" SET??
 11463  056466  001024                         BNE     10$             ;YES-DONT REPORT "MOL" = 0
 11464  056470  013737  001342  001140         MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
 11465  056476  052737  010000  001140         BIS     #MOL,$GDDAT
 11466  056504  013737  001342  001142         MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
 11467  056512  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 11468  056516  112776  000207  000000         MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
 11469  056524  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11470  056530  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 11471  056532  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
11472  056536  000240                          NOP
11473  056540                          10$:
11474
11475                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11476  056540  032737  000100  001342          BIT     #VV,RMDSI       ;IS "VV" = 0??
11477  056546  001030                          BNE     20$             ;NO!!
11478  056550  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
11479  056556  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
11480  056560  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11481  056566  052737  000100  001342          BIS     #VV,RMDSI
11482  056574  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11483  056602  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11484  056606  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
11485  056614  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11486  056620  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11487  056622  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11488  056626  000240                          NOP
11489  056630                          20$:
11490
11491                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11492  056630  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11493  056636  001430                          BEQ     30$             ;NO!!
11494  056640  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11495  056646  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
11496  056650  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11497  056656  042737  020000  001140          BIC     #PIP,$GDDAT
11498  056664  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11499  056672  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11500  056676  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11501  056704  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11502  056710  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11503  056712  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11504  056716  000240                          NOP
11505  056720                          30$:
11506
11507                                  ;SEE IF "SKI" = "DVC" = 0
11508  056720  013746  001372                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
11509  056724  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
11510  056730  001460                          BEQ     60$             ;BRANCH IF NO ERROR
11511  056732                          40$:
11512
11513                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11514  056732  032737  000200  001372          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
11515  056740  001424                          BEQ     50$             ;NO!!
11516  056742  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11517  056750  042737  000200  001140          BIC     #DVC,$GDDAT
11518  056756  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11519  056764  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
11520  056770  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11521  056776  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11522  057002  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11523  057004  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11524  057010  000240                          NOP
11525  057012                          50$:
11526
11527                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
11528  057012  032737  040000  001372          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
11529  057020  001424                          BEQ     60$             ;NO!!
11530  057022  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11531  057030  042737  040000  001140          BIC     #SKI,$GDDAT
11532  057036  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11533  057044  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11534  057050  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
11535  057056  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11536  057062  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11537  057064  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11538  057070  000240                          NOP
11539  057072                          60$:
11540
11541                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11542  057072  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11543  057076  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11544  057102  001403                          BEQ     70$             ;NO!!
11545  057104  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
11546  057110  000402                          BR      80$
11547  057112  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
11548  057116  000240                  80$:    NOP
11549  057120  000207                          RTS     PC              ;RETURN TO USER
```

```
11550                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
11551
11552                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
11553                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
11554                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
11555                                   ;THE MASKS ARE APPLIED.
11556
11557                                   ;CALL:
11558                                   ;(1)    JSR     PC,CMPERRSTS
11559                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
11560                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
11561                                   ;       BR      ???             RETURN HERE IF NO ERROR
11562                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11563                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11564                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11565                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11566
11567                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
11568                                   ;BE ZERO
11569
11570  057122                           CMPERRSTS:
11571
11572                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
11573  057122  013737  001344  001176           MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
11574  057130  047637  000000  001176           BIC     @(SP),$TMP1     ;MASK   RMER1
11575  057136  062716  000002                   ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
11576  057142  013737  001372  001200           MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
11577  057150  047637  000000  001200           BIC     @(SP),$TMP2     ;MASK RMER2
11578
11579
11580                                   ;CLEAR USER'S ERROR CALL
11581  057156  062716  000006                   ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
11582  057162  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
11583  057166  162716  000004                   SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
11584
11585                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
11586  057172  005737  001176                   TST     $TMP1           ;ANY ERRORS TO REPORT??
11587  057176  001420                           BEQ     5$              ;NO !!
11588  057200  013737  001176  001142           MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
11589  057206  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
11590  057212  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11591  057216  112776  000066  000000           MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
11592  057224  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11593  057230  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11594  057232  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11595  057236  000240                           NOP
11596  057240                           5$:
11597
11598
11599                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
11600  057240  005737  001200                   TST     $TMP2           ;ANY ERRORS IN RMER2?
11601  057244  001420                           BEQ     10$             ;NO!!
11602
11603  057246  013737  001200  001142           MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
11604  057254  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
11605  057260  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11606  057264  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11607  057272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11608  057276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11609  057300  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11610  057304  000240                          NOP
11611  057306                          10$:
11612
11613                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11614  057306  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11615  057312  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
11616  057316  001403                          BEQ     20$             ;NO!!
11617  057320  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11618  057324  000402                          BR      30$
11619  057326  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11620  057332  000207                  30$:    RTS     PC              ;RETURN TO USER
11621
11622
```

```
                                        .SBTTL  STOP AND SHUTDOWN SUBROUTINES

                                        STOP:

                                        ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
                                                MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
                                                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
                                                RTI                     ;;POP NEW PC AND PS
                                        64$:
                                                NOP
                                        ;RAISE PRIORITY TO INHIBIT INTERRUPT
                                                MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
                                                MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
                                                RTI                     ;;POP NEW PC AND PS
                                        65$:

                                        10$:    RTS     PC              ;CONTINUE


                                        SHUT:
                                                TST     @#CTLFG         ;HALT ?
                                                BNE     .+6             ;BRANHC IF YES
                                                JMP     READY           ;CONTINUE
                                                TYPE    ,100$           ;TYPE THE HALT MESSAGE
                                                TST     42              ;RUNNING STANDALONE ??
                                                BEQ     10$             ;YES !!
                                                JMP     $EOP            ;NO - GO TO END OF PASS
                                        10$:
                                                JMP     START           ;GO TO START
                                        100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>

                                                .EVEN
```

```
11657  057444  012737  177777  001326  SHUT2:  MOV     #-1,CTLFG               ;SET THE CONTROL-C FLAG
11658  057452  000002                          RTI                             ;EXIT FROM INTERRUPT
11659                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11660
11661                                  ;;*****************************************************************
11662                                  ;*SAVE R0-R5
11663                                  ;*CALL:
11664                                  ;*      SAVREG
11665                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11666                                  ;*
11667                                  ;*TOP---(+16)
11668                                  ;*  +2---(+18)
11669                                  ;*  +4---R5
11670                                  ;*  +6---R4
11671                                  ;*  +8---R3
11672                                  ;*+10---R2
11673                                  ;*+12---R1
11674                                  ;*+14---R0
11675
11676  057454                          $SAVREG:
11677  057454  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11678  057456  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11679  057460  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11680  057462  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11681  057464  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11682  057466  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11683  057470  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11684  057474  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11685  057500  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11686  057504  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11687  057510  000002                          RTI
11688
11689                                  ;*RESTORE R0-R5
11690                                  ;*CALL:
11691                                  ;*      RESREG
11692  057512                          $RESREG:
11693  057512  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11694  057516  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11695  057522  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11696  057526  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11697  057532  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11698  057534  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11699  057536  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11700  057540  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11701  057542  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11702  057544  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11703  057546  000002                          RTI
11704                                  .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
11705
11706                                  ;;*****************************************************************
11707                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
11708                                  ;*BINARY-ASCII NUMBER AND TYPE IT.
11709                                  ;*CALL:
11710                                  ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
11711                                  ;*      TYPBN                   ;;TYPE IT
11712
```

```
11713  057550  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11714  057552  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
11715  057556  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
11716  057560  112737  000060  057622  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
11717  057566  006101                          ROL     R1              ;;GET THIS BIT
11718  057570  001406                          BEQ     2$              ;;DONE?
11719  057572  105537  057622                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
11720  057576  104401  057622                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
11721  057602  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
11722  057604  000765                          BR      1$              ;;GO DO THE NEXT BIT
11723  057606  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
11724  057610  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11725  057616  012616                          MOV     (SP)+,(SP)
11726  057620  000002                          RTI                     ;;RETURN TO USER
11727  057622      000     000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
11728                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11729
11730                                  ;;**************************************************************
11731                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11732                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11733                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11734                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11735                                  ;*REPLACED WITH SPACES.
11736                                  ;*CALL:
11737                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
11738                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
11739
11740  057624                          $TYPDS:
11741  057624  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11742  057626  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11743  057630  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11744  057632  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11745  057634  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11746  057636  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
11747  057642  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
11748  057646  100004                          BPL     1$              ;;BR IF INPUT IS POS.
11749  057650  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
11750  057652  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
11751  057660  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
11752  057662  012703  060040                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
11753  057666  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
11754  057672  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
11755  057674  016001  060030                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
11756  057700  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
11757  057702  002402                          BLT     4$              ;;BR IF DONE
11758  057704  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
11759  057706  000774                          BR      3$
11760  057710  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
11761  057712  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
11762  057714  001002                          BNE     5$              ;;FALL THROUGH IF 0
11763  057716  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
11764  057720  100407                          BMI     7$              ;;BR IF YES
11765  057722  106316                  5$:     ASLB    (SP)            ;;MSD?
11766  057724  103003                          BCC     6$              ;;BR IF NO
11767  057726  116363  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
11768  057734  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
```

```
11769  057740  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11770  057744  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11771  057746  005720                          TST     (R0)+           ;;JUST INCREMENTING
11772  057750  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
11773  057754  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
11774  057756  003002                          BGT     8$              ;;GO TO EXIT
11775  057760  010502                          MOV     R5,R2           ;;GET THE LSD
11776  057762  000764                          BR      6$              ;;GO CHANGE TO ASCII
11777  057764  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
11778  057766  100003                          BPL     9$              ;;BR IF NO
11779  057770  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11780  057776  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
11781  060000  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11782  060002  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11783  060004  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11784  060006  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11785  060010  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11786  060012  104401  060040                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
11787  060016  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11788  060024  012616                          MOV     (SP)+,(SP)
11789  060026  000002                          RTI                     ;;RETURN TO USER
11790  060030  023420                  $DTBL:  10000.
11791  060032  001750                          1000.
11792  060034  000144                          100.
11793  060036  000012                          10.
11794  060040  000004                  $DBLK:  .BLKW   4
11795                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11796
11797                                  ;;**************************************************************
11798                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11799                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11800                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11801                                  ;*CALL:
11802                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11803                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
11804                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11805                                  ;*      .BYTE   M               ;;M=1 OR 0
11806                                  ;*                                      ;;1=TYPE LEADING ZEROS
11807                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
11808                                  ;*
11809                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11810                                  ;*$TYPOS OR $TYPOC
11811                                  ;*CALL:
11812                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11813                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
11814                                  ;*
11815                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11816                                  ;*CALL:
11817                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11818                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
11819
11820  060050  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
11821  060054  116637  000001  060273          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
11822  060062  112637  060275                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
11823  060066  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
11824  060072  000406                          BR      $TYPON
```

```
11825  060074  112737  000001  060273  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
11826  060102  112737  000006  060275          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
11827  060110  112737  000005  060272  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
11828  060116  010346                          MOV     R3,-(SP)        ;;SAVE R3
11829  060120  010446                          MOV     R4,-(SP)        ;;SAVE R4
11830  060122  010546                          MOV     R5,-(SP)        ;;SAVE R5
11831  060124  113704  060275                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
11832  060130  005404                          NEG     R4
11833  060132  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11834  060136  110437  060274                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
11935  060142  113704  060273                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11836  060146  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11837  060152  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
11838  060154  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
11839  060156  000404                  BR      3$              ;;GO DO MSB
11840  060160  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
11841  060162  006105                  ROL     R5
11842  060164  006105                  ROL     R5
11843  060166  010503                  MOV     R5,R3
11844  060170  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
11845  060172  105337  060274          DECB    $OMODE          ;;TYPE THIS DIGIT?
11846  060176  100016                  BPL     7$              ;;BR IF NO
11847  060200  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
11848  060204  001002                  BNE     4$              ;;TEST FOR 0
11849  060206  005704                  TST     R4              ;;SUPPRESS THIS 0?
11850  060210  001403                  BEQ     5$              ;;BR IF YES
11851  060212  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
11852  060214  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11853  060220  052703  000040  5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
11854  060224  110337  060270          MOVB    R3,8$           ;;SAVE FOR TYPING
11855  060230  104401  060270          TYPE    ,8$             ;;GO TYPE THIS DIGIT
11856  060234  105337  060272  7$:     DECB    $OCNT           ;;COUNT BY 1
11857  060240  003347                  BGT     2$              ;;BR IF MORE TO DO
11858  060242  002402                  BLT     6$              ;;BR IF DONE
11859  060244  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11860  060246  000744                  BR      2$              ;;GO DO THE LAST DIGIT
11861  060250  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
11862  060252  012604                  MOV     (SP)+,R4        ;;RESTORE R4
11863  060254  012603                  MOV     (SP)+,R3        ;;RESTORE R3
11864  060256  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
11865  060264  012616                  MOV     (SP)+,(SP)
11866  060266  000002                  RTI                     ;;RETURN
11867  060270     000  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
11868  060271     000          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
11869  060272     000  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
11870  060273     000  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
11871  060274  000000  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
11872                  .SBTTL  TYPE ROUTINE
11873
11874                  ;;*****************************************************************
11875                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11876                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11877                  ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11878                  ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11879                  ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11880                  ;*
```

```
11881                                   ;*CALL:
11882                                   ;*1) USING A TRAP INSTRUCTION
11883                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11884                                   ;*OR
11885                                   ;*      TYPE
11886                                   ;*      MESADR
11887                                   ;*
11888
11889  060276  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
11890  060302  100002                          BPL     1$              ;;BR IF YES
11891  060304  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
11892  060306  000430                          BR      3$              ;;LEAVE
11893  060310  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
11894  060312  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
11895  060316  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11896  060324  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
11897  060326  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
11898  060334  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
11899  060336  010037  060346                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
11900  060342  004737  063222                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
11901  060346  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
11902  060350  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
11903  060356  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
11904  060360  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
11905  060362  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
11906  060364  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
11907  060366  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
11908  060370  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
11909  060374  000002                          RTI                     ;;RETURN
11910  060376  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
11911  060402  001430                          BEQ     8$
11912  060404  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
11913  060410  001006                          BNE     5$
11914  060412  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
11915  060414  104401                          TYPE                    ;;TYPE A CR AND LF
11916  060416  001217                          $CRLF
11917  060420  105037  060554                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
11918  060424  000755                          BR      2$              ;;GET NEXT CHARACTER
11919  060426  004737  060510          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
11920  060432  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
11921  060436  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
11922  060440  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
11923                                                                  ;;AND THE NULL CHAR.
11924  060444  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
11925  060450  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
11926  060452  004737  060510                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
11927  060456  105337  060554                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
11928  060462  000770                          BR      7$              ;;LOOP
11929
11930                                   ;HORIZONTAL TAB PROCESSOR
11931
11932  060464  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
11933  060470  004737  060510          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
11934  060474  132737  000007  060554          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
11935  060502  001372                          BNE     9$              ;;TAB STOP
11936  060504  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
```

```
11937  060506  000724                          BR      2$              ;;GET NEXT CHARACTER
11938  060510  105777  120450          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
11939  060514  100375                          BPL     $TYPEC
11940  060516  116677  000002  120442          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11941  060524  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
11942  060532  001003                          BNE     1$              ;;BRANCH IF NO
11943  060534  105037  060554                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
11944  060540  000406                          BR      $TYPEX          ;;EXIT
11945  060542  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
11946  060550  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
11947  060552  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
11948  060554  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
11949  060556  000207          $TYPEX: RTS     PC
11950
11951                                  .SBTTL  SCOPE HANDLER ROUTINE
11952
11953                                  ;;*****************************************************************
11954                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11955                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11956                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11957                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11958                                  ;*SW14=1        LOOP ON TEST
11959                                  ;*SW11=1        INHIBIT ITERATIONS
11960                                  ;*SW09=1        LOOP ON ERROR
11961                                  ;*SW08=1        LOOP ON TEST IN SWR<7:0>
11962                                  ;*CALL
11963                                  ;*      SCOPE           ;;SCOPE=IOT
11964
11965  060560                          $SCOPE:
11966  060560  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
11967  060562  004737  057334                  JSR     PC,STOP
11968  060566  032777  040000  120360  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11969  060574  001131                          BNE     $OVER           ;;YES IF SW14=1
11970                                  ;*****START OF CODE FOR THE XOR TESTER*****
11971  060576  000416          $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11972                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11973  060600  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11974  060604  012737  060624  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11975  060612  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
11976  060616  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11977  060622  000500                          BR      $SVLAD          ;;GO TO THE NEXT TEST
11978  060624  022626          5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
11979  060626  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11980  060632  000440                          BR      7$              ;;LOOP ON THE PRESENT TEST
11981  060634                          6$:;*****END OF CODE FOR THE XOR TESTER*****
11982  060634  032777  000400  120312          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
11983  060642  001421                          BEQ     2$              ;;BR IF NO
11984  060644  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
11985  060646  117716  120302                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
11986  060652  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
11987  060654  022716  000035                  CMP     #35,(SP)        ;;CHECK THE NUMBER IN THE SWR
11988  060660  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
11989  060662  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
11990  060666  005316                          DEC     (SP)            ;;BACKUP BY ONE
11991  060670  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
11992  060672  062716  061076                  ADD     #$SWOBTBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
```

```
11993  060676  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
11994  060702  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
11995  060704  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
11996  060706  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11997  060712  001421                          BEQ     3$              ;;BR IF NO
11998  060714  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
11999  060722  101015                          BHI     3$              ;;BR IF NO
12000  060724  032777  001000  120222          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
12001  060732  001404                          BEQ     4$              ;;BR IF NO
12002  060734  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
12003  060742  000446                          BR      $OVER
12004  060744  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
12005  060750  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
12006  060754  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
12007  060756  032777  004000  120170  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
12008  060764  001011                          BNE     1$              ;;BR IF YES
12009  060766  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
12010  060772  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
12011  060774  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
12012  061000  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
12013  061006  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
12014  061010  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
12015  061016  013737  061074  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
12016  061024  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
12017  061030  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
12018  061036  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
12019  061042  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
12020  061046  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
12021  061052  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
12022  061060  013777  001116  120070  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
12023  061066  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
12024  061072  000002                          RTI                     ;;FIXES PS
12025  061074  000012          $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
12026  061076                  $SWOBTBL:
12027  061076  007454                  .WORD   TST1+2                  ;;STARTING ADDRESS OF TEST 1
12028  061100  007362                  .WORD   TST2+2                  ;;STARTING ADDRESS OF TEST 2
12029  061102  007546                  .WORD   TST3+2                  ;;STARTING ADDRESS OF TEST 3
12030  061104  007730                  .WORD   TST4+2                  ;;STARTING ADDRESS OF TEST 4
12031  061106  010530                  .WORD   TST5+2                  ;;STARTING ADDRESS OF TEST 5
12032  061110  011330                  .WORD   TST6+2                  ;;STARTING ADDRESS OF TEST 6
12033  061112  012144                  .WORD   TST7+2                  ;;STARTING ADDRESS OF TEST 7
12034  061114  012720                  .WORD   TST10+2                 ;;STARTING ADDRESS OF TEST 10
12035  061116  013640                  .WORD   TST11+2                 ;;STARTING ADDRESS OF TEST 11
12036  061120  014440                  .WORD   TST12+2                 ;;STARTING ADDRESS OF TEST 12
12037  061122  015214                  .WORD   TST13+2                 ;;STARTING ADDRESS OF TEST 13
12038  061124  016132                  .WORD   TST14+2                 ;;STARTING ADDRESS OF TEST 14
12039  061126  016706                  .WORD   TST15+2                 ;;STARTING ADDRESS OF TEST 15
12040  061130  017462                  .WORD   TST16+2                 ;;STARTING ADDRESS OF TEST 16
12041  061132  020236                  .WORD   TST17+2                 ;;STARTING ADDRESS OF TEST 17
12042  061134  021040                  .WORD   TST20+2                 ;;STARTING ADDRESS OF TEST 20
12043  061136  021564                  .WORD   TST21+2                 ;;STARTING ADDRESS OF TEST 21
12044  061140  022310                  .WORD   TST22+2                 ;;STARTING ADDRESS OF TEST 22
12045  061142  023032                  .WORD   TST23+2                 ;;STARTING ADDRESS OF TEST 23
12046  061144  023330                  .WORD   TST24+2                 ;;STARTING ADDRESS OF TEST 24
12047  061146  023626                  .WORD   TST25+2                 ;;STARTING ADDRESS OF TEST 25
12048  061150  024354                  .WORD   TST26+2                 ;;STARTING ADDRESS OF TEST 26
```

```
061152  025102                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
061154  025630                          .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
061156  026522                          .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
061160  027330                          .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
061162  030052                          .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
061164  030574                          .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
061166  031752                          .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

061170                          $ERROR:
061170  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
061172  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
061176  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
061200  013777  001116  117750          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
061206  032777  002000  117740          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
061214  001402                          BEQ     1$              ;;NO - SKIP
061216  104401  001212                  TYPE    ,$BELL          ;;RING BELL
061222  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
061226  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
061232  162737  000002  001132          SUB     #2,$ERRPC
061240  117737  117666  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
061246  032777  020000  117700          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
061254  001004                          BNE     20$             ;;SKIP TYPEOUTS
061256  004737  033176                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
061262  104401  001217                  TYPE    ,$CRLF
061266                          20$:
061266  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
061274  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
061276  113737  001130  061310          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
061304  004737  063232                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
061310  000                     21$:    .BYTE   0
061311  000                             .BYTE   0
061312  000777                  22$:    BR      22$             ;;APT ERROR LOOP
061314  005777  117634          2$:     TST     @SWR            ;;HALT ON ERROR
061320  100002                          BPL     3$              ;;SKIP IF CONTINUE
061322  000000                          HALT                    ;;HALT ON ERROR!
061324  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
061326  032777  001000  117620  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
061334  001402                          BEQ     4$              ;;BR IF NO
061336  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
061342  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
061346  001402                          BEQ     5$              ;;BR IF NONE
061350  013716  001210                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
061354                          5$:
```

```
                                ;ERROR HANDLER ROUTINE
12105  061354  022737  033156  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
12106  061362  001001                          BNE     6$              ;;BRANCH IF NO
12107  061364  000000                          HALT                    ;;YES
12108  061366                          6$:
12109  061366  000002                          RTI                     ;;RETURN
12110                           .SBTTL  TTY INPUT ROUTINE
12111
12112                           ;;******************************************************************
12113                           .ENABL  LSB
12114  061370  000000          $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
12115  061372  000000          $TKQIN: .WORD   0               ;;INPUT POINTER
12116  061374  000000          $TKQOUT: .WORD  0               ;;OUTPUT POINTER
12117  061376  000001          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
12118           061377          $TKQEND=.
12119           061400          .EVEN
12120
12121                           ;*TK INITIALIZE ROUTINE
12122                           ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12123                           ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12124                           ;
12125                           ;*CALL:
12126                           ;*      JSR     PC,$TKINT
12127                           ;*      RETURN
12128                           ;
12129  061400  005037  061370  $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
12130  061404  012737  061376  061372          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
12131  061412  013737  061372  061374          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12132  061420  012737  061450  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
12133  061426  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
12134  061434  005777  117522                  TST     @$TKB           ;;CLEAR DONE FLAG
12135  061440  012777  000100  117512          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
12136  061446  000207                          RTS     PC              ;;RETURN TO CALLER
12137
12138                           ;*TK SERVICE ROUTINE
12139                           ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12140                           ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
12141                           ;*IT IN THE QUEUE.
12142                           ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
12143                           ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT2)
12144                           ;
12145  061450  117746  117506  $TKSRV: MOVB     @$TKB,-(SP)     ;;PICKUP THE CHARACTER
12146  061454  042716  177600          BIC     #^C177,(SP)     ;;STRIP THE JUNK
12147  061460  021627  000003          CMP     (SP),#3         ;;IS IT A CONTROL C?
12148  061464  001007                  BNE     1$              ;;BRANCH IF NO
12149  061466  104401  062564          TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
12150  061472  004737  061400          JSR     PC,$TKINT       ;;INIT THE KEYBOARD
12151  061476  005726                  TST     (SP)+           ;;CLEAN UP STACK
12152  061500  000137  057444          JMP     SHUT2           ;;CONTROL C RESTART
12153  061504  021627  000007  1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
12154  061510  001004                  BNE     2$              ;;BRANCH IF NO
12155  061512  022737  000176  001154  CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
12156  061520  001500                  BEQ     6$              ;;GO TO SWR CHANGE
12157
12158  061522                  2$:
12159  061522  022737  000001  061370  CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
12160  061530  001004                  BNE     3$              ;;BRANCH IF NO
```

```
12161  061532  104401  001212                  TYPE    $BELL           ;;RING THE TTY BELL
12162  061536  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
12163  061540  000451                          BR      5$              ;;EXIT
12164  061542  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
12165  061546  001021                          BNE     32$             ;;BRANCH IF NO
12166  061550  005077  117404                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12167  061554  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
12168  061556  105777  117376          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
12169  061562  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
12170  061564  117746  117372                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
12171  061570  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12172  061574  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
12173  061600  001366                          BNE     31$             ;;BRANCH IF NO
12174  061602  012777  000100  117350          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
12175  061610  000002                          RTI                     ;;RETURN
12176  061612  005237  061370          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
12177  061616  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
12178  061622  002405                          BLT     4$              ;;BRANCH IF YES
12179  061624  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
12180  061630  003002                          BGT     4$              ;;BRANCH IF YES
12181  061632  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
12182  061636  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
12183  061642  005237  061372                  INC     $TKQIN          ;;UPDATE THE POINTER
12184  061646  023727  061372  061377          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
12185  061654  001003                          BNE     5$              ;;BRANCH IF NO
12186  061656  012737  061376  061372          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
12187  061664  000002                  5$:     RTI                     ;;RETURN
12188
12189                                          ;;**************************************************************
12190                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12191                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
12192                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12193                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12194  061666  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
12195  061674  001124                          BNE     1S$             ;;EXIT IF NOT
12196  061676  105777  117256                  TSTB    @$TKS           ;;IS A CHAR WAITING?
12197  061702  100121                          BPL     1S$             ;;IF NOT, EXIT
12198  061704  117746  117252                  MOVB    @$TKB,-(SP)     ;;YES
12199  061710  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
12200  061714  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
12201  061720  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
12202                                                                  ;;AND EXIT
12203
12204                                          ;;**************************************************************
12205                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12206                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12207                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12208  061722  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
12209  061730  001674                          BEQ     2$              ;;BRANCH IF YES
12210  061732  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
12211  061734  004737  061400                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
12212  061740  005077  117214                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12213  061744  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
12214
12215  061752  104401  062576                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
12216  061756  104401  062603          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
```

```
12217  061762  013746  000176            MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
12218  061766  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
12219  061770  104401  062614            TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
12220  061774  005046            19$:    CLR     -(SP)           ;;CLEAR COUNTER
12221  061776  005046                    CLR     -(SP)           ;;THE NEW SWR
12222  062000  105777  117154      7$:   TSTB    @$TKS           ;;CHAR THERE?
12223  062004  100375                    BPL     7$              ;;IF NOT TRY AGAIN
12224
12225  062006  117746  117150            MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
12226  062012  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12227
12228  062016  021627  000003            CMP     (SP),#3         ;;IS IT A CONTROL-C?
12229  062022  001015                    BNE     9$              ;;BRANCH IF NOT
12230  062024  104401  062564            TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
12231  062030  062706  000006            ADD     #6,SP           ;;CLEAN UP STACK
12232  062034  123727  001151  000001    CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
12233  062042  001003                    BNE     8$              ;;BRANCH IF NO
12234  062044  012777  000100  117106    MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
12235  062052  000137  057444      8$:   JMP     SHUT2           ;;CONTROL-C RESTART
12236
12237
12238  062056  021627  000025      9$:   CMP     (SP),#25        ;;IS IT A CONTROL-U?
12239  062062  001005                    BNE     10$             ;;BRANCH IF NOT
12240  062064  104401  062571            TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
12241  062070  062706  000006      20$:  ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
12242  062074  000737                    BR      19$             ;;LET'S TRY IT AGAIN
12243
12244
12245  062076  021627  000015      10$:  CMP     (SP),#15        ;;IS IT A <CR>?
12246  062102  001022                    BNE     16$             ;;BRANCH IF NO
12247  062104  005766  000004            TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
12248  062110  001403                    BEQ     11$             ;;BRANCH IF YES
12249  062112  016677  000002  117034    MOV     2(SP),@SWR      ;;SAVE NEW SWR
12250  062120  062706  000006      11$:  ADD     #6,SP           ;;CLEAR UP STACK
12251  062124  104401  001217      14$:  TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
12252  062130  123727  001151  000001    CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
12253  062136  001003                    BNE     15$             ;;BRANCH IF NOT
12254  062140  012777  000100  117012    MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
12255  062146  000002            15$:    RTI                     ;;RETURN
12256  062150  004737  060510      16$:  JSR     PC,$TYPEC       ;;ECHO CHAR
12257  062154  021627  000060            CMP     (SP),#60        ;;CHAR < 0?
12258  062160  002420                    BLT     18$             ;;BRANCH IF YES
12259  062162  021627  000067            CMP     (SP),#67        ;;CHAR > 7?
12260  062166  003015                    BGT     18$             ;;BRANCH IF YES
12261  062170  042726  000060            BIC     #60,(SP)+       ;;STRIP-OFF ASCII
12262  062174  005766  000002            TST     2(SP)           ;;IS THIS THE FIRST CHAR
12263  062200  001403                    BEQ     17$             ;;BRANCH IF YES
12264  062202  006316                    ASL     (SP)            ;;NO, SHIFT PRESENT
12265  062204  006316                    ASL     (SP)            ;;    CHAR OVER TO MAKE
12266  062206  006316                    ASL     (SP)            ;;    ROOM FOR NEW ONE.
12267  062210  005266  000002      17$:  INC     2(SP)           ;;KEEP COUNT OF CHAR
12268  062214  056616  177776            BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
12269  062220  000667                    BR      7$              ;;GET THE NEXT ONE
12270  062222  104401  001216      18$:  TYPE    ,$QUES          ;;TYPE ?<CR><LF>
12271  062226  000720                    BR      20$             ;;SIMULATE CONTROL-U
12272                             .DSABL  LSB
```

```
12273
12274
12275                                   ;;**********************************************************************
12276                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
12277                                   ;*CALL:
12278                                   ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
12279                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
12280                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
12281                                   ;
12282
12283  062230  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
12284  062232  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
12285  062240  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
12286  062244  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
12287  062246  012746  062254                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12288  062252  000002                          RTI                     ;;POP NEW PC AND PS
12289  062254                          64$:
12290  062254  005737  061370          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
12291  062260  001775                          BEQ     1$
12292  062262  005337  061370                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
12293  062266  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
12294  062274  005237  061374                  INC     $TKQOUT         ;;UPDATE THE POINTER
12295  062300  023727  061374  061377          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
12296  062306  001003                          BNE     2$              ;;BRANCH IF NO
12297  062310  012737  061376  061374          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
12298  062316  000002                  2$:     RTI                     ;;RETURN
12299                                   ;;**********************************************************************
12300                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
12301                                   ;*CALL:
12302                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
12303                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
12304                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
12305
12306  062320  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
12307  062322  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
12308  062324  012703  062554          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
12309  062330  022703  062564          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
12310  062334  101456                          BLOS    4$              ;;BR IF YES
12311  062336  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
12312  062340  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
12313  062342  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
12314  062346  001022                          BNE     5$              ;;BR IF NO
12315  062350  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
12316  062352  001007                          BNE     6$              ;;BR IF NO
12317  062354  112737  000134  062552          MOVB    #'\,9$          ;;TYPE A BACK SLASH
12318  062362  104401  062552                  TYPE    ,9$
12319  062366  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
12320  062372  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
12321  062374  020327  062554                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
12322  062400  103434                          BLO     4$              ;;BR IF YES
12323  062402  111337  062552                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
12324  062406  104401  062552                  TYPE    ,9$             ;;GO TYPE
12325  062412  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
12326  062414  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
12327  062416  001406                          BEQ     7$              ;;BR IF NO
12328  062420  112737  000134  062552          MOVB    #'\,9$          ;;TYPE A BACK SLASH
```

```
12329  062426  104401  062552                  TYPE    #9$
12330  062432  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
12331  062434  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
12332  062440  001003                          BNE     8$              ;;BR IF NO
12333  062442  104401  062571                  TYPE    $CNTLU          ;;TYPE A CONTROL "U"
12334  062446  000726                          BR      1$              ;;GO START OVER
12335  062450  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
12336  062454  001011                          BNE     3$              ;;BRANCH IF NO
12337  062456  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
12338  062460  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
12339  062464  104401  062554                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
12340  062470  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
12341  062472  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
12342  062476  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
12343  062500  111337  062552          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
12344  062504  104401  062552                  TYPE    ,9$
12345  062510  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
12346  062514  001305                          BNE     2$              ;;LOOP IF NOT RETURN
12347  062516  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
12348  062522  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
12349  062526  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
12350  062530  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12351  062532  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
12352  062534  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
12353  062542  012766  062554  000004          MOV     #$TTYIN,4(SP)
12354  062550  000002                          RTI                     ;;RETURN
12355  062552     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
12356  062553     000                  .BYTE   0                       ;;TERMINATOR
12357  062554  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
12358  062564  041536  005015   000    $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
12359  062571     136  006525  000012 $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
12360  062576  043536  005015   000    $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
12361  062603     015  051412  051127 $MSWR:  .ASCIZ  <15><12>/SWR = /
12362  062610  036440  000040
12363  062614  020040  042516  020127 $MNEW:  .ASCIZ  /  NEW = /
12364  062622  020075     000
12365          062626                  .EVEN
12366                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
12367
12368                                  ;;*****************************************************************
12369                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
12370                                  ;*CHANGE IT TO BINARY.
12371                                  ;*CALL:
12372                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
12373                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
12374                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
12375
12376  062626  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
12377  062630  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
12378  062636  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12379  062640  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12380  062642  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12381  062644  104412          1$:     RDLIN                   ;;READ AN ASCIZ LINE
12382  062646  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
12383  062650  005001                  CLR     R1              ;;CLEAR DATA WORD
12384  062652  005002                  CLR     R2
```

```
                                READ AN OCTAL NUMBER FROM THE TTY

12385  062654  112046           2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
12386  062656  001412                   BEQ     3$              ;;IF ZERO GET OUT
12387  062660  006301                   ASL     R1              ;;*2
12388  062662  006102                   ROL     R2
12389  062664  006301                   ASL     R1              ;;*4
12390  062666  006102                   ROL     R2
12391  062670  006301                   ASL     R1              ;;*8
12392  062672  006102                   ROL     R2
12393  062674  042716  177770           BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
12394  062700  062601                   ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
12395  062702  000764                   BR      2$              ;;LOOP
12396  062704  005726           3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
12397  062706  010166  000012           MOV     R1,12(SP)       ;;SAVE THE RESULT
12398  062712  010237  062726           MOV     R2,$HIOCT
12399  062716  012602                   MOV     (SP)+,R2        ;;POP STACK INTO R2
12400  062720  012601                   MOV     (SP)+,R1        ;;POP STACK INTO R1
12401  062722  012600                   MOV     (SP)+,R0        ;;POP STACK INTO R0
12402  062724  000002                   RTI                     ;;RETURN
12403  062726  000000           $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
12404                            .SBTTL  TRAP DECODER
12405
12406                            ;;**************************************************************************
12407                            ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12408                            ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12409                            ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12410                            ;*GO TO THAT ROUTINE.
12411
12412  062730  016646  000002   $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
12413  062734  042716  000020           BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
12414  062740  012746  062746           MOV     #1$,-(SP)       ;;  T-BIT TRAPS
12415  062744  000002                   RTI                     ;;SET THE NEW STATUS
12416  062746  010046           1$:     MOV     R0,-(SP)        ;;SAVE R0
12417  062750  016600  000002           MOV     2(SP),R0        ;;GET TRAP ADDRESS
12418  062754  005740                   TST     -(R0)           ;;BACKUP BY 2
12419  062756  111000                   MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
12420  062760  006300                   ASL     R0              ;;POSITION FOR INDEXING
12421  062762  016000  063002           MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
12422  062766  000200                   RTS     R0              ;;GO TO ROUTINE
12423
12424
12425                            ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12426
12427  062770  011646           $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
12428  062772  016666  000004  000002   MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
12429  063000  000002                   RTI                     ;;RESTORE THE PSW
12430
12431                            .SBTTL  TRAP TABLE
12432
12433                            ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12434                            ;*BY THE "TRAP" INSTRUCTION.
12435
12436                            ;       ROUTINE
12437                            ;       -------
12438  063002  062770           $TRPAD: .WORD   $TRAP2
12439  063004  060276                   $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
12440  063006  060074                   $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
```

```
12441  063010  060050                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12442  063012  060110                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
12443  063014  057624                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12444  063016  057550                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
12445
12446  063020  061756                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
12447
12448  063022  061666                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
12449  063024  062230                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
12450  063026  062320                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
12451  063030  062626                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
12452  063032  057454                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
12453  063034  057512                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
12454                                  .SBTTL  POWER DOWN AND UP ROUTINES
12455
12456                                  ;;**************************************************************
12457                                  ;POWER DOWN ROUTINE
12458  063036  012737  063176  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12459  063044  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
12460  063052  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
12461  063054  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
12462  063056  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
12463  063060  010346                          MOV     R3,-(SP)         ;;PUSH R3 ON STACK
12464  063062  010446                          MOV     R4,-(SP)         ;;PUSH R4 ON STACK
12465  063064  010546                          MOV     R5,-(SP)         ;;PUSH R5 ON STACK
12466  063066  017746  116062                  MOV     @$WR,-(SP)       ;;PUSH @SWR ON STACK
12467  063072  010637  063202                  MOV     SP,$SAVR6        ;;SAVE SP
12468  063076  012737  063110  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12469  063104  000000                          HALT
12470  063106  000776                          BR      .-2              ;;HANG UP
12471
12472                                  ;;**************************************************************
12473                                  ;POWER UP ROUTINE
12474  063110  012737  063176  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12475  063116  013706  063202                  MOV     $SAVR6,SP        ;;GET SP
12476  063122  005037  063202                  CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
12477  063126  005237  063202          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
12478  063132  001375                          BNE     1$               ;;OF   WORD
12479  063134  012677  116014                  MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
12480  063140  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
12481  063142  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
12482  063144  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
12483  063146  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
12484  063150  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
12485  063152  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
12486  063154  012737  063036  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12487  063162  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
12488  063170  104401                          TYPE                     ;REPORT THE POWER FAILURE
12489  063172  063204                  $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
12490  063174  000002                          RTI
12491  063176  000000                  $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
12492  063200  000776                          BR      .-2              ;;  BEFORE THE POWER DOWN WAS COMPLETE
12493  063202  000000                  $SAVR6: 0                        ;;PUT THE SP HERE
12494  063204  005015  047520  042527  $POWER: .ASCIZ   <15><12>"POWER"
12495  063212  000122
12496                                          .EVEN
```

```
12497                                   .SBTTL  APT COMMUNICATIONS ROUTINE
12498
12499                                   ;;*****************************************************************
12500  063214  112737  000001  063460  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
12501  063222  112737  000001  063456  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
12502  063230  000403                  BR      $ATYC
12503  063232  112737  000001  063460  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
12504  063240                          $ATYC:
12505  063240  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12506  063242  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12507  063244  105737  063456                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
12508  063250  001450                          BEQ     5$              ;;IF NOT:  BR
12509  063252  122737  000001  001242          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
12510  063260  001031                          BNE     3$              ;;IF NOT:  BR
12511  063262  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
12512  063270  001425                          BEQ     3$              ;;IF NOT:  BR
12513  063272  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
12514  063276  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12515  063304  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
12516  063310  001375                          BNE     1$              ;;IF NOT:  WAIT
12517  063312  010037  001236                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
12518  063316  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
12519  063320  001376                          BNE     2$
12520  063322  163700  001236                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
12521  063326  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
12522  063330  010037  001240                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
12523  063334  012737  000004  001222          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
12524  063342  000413                          BR      5$
12525  063344  017637  000004  063370  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
12526  063352  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
12527  063360  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
12528  063364  004737  060276                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
12529  063370  000000                  4$:     .WORD   0
12530  063372                          5$:
12531  063372  105737  063460          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
12532  063376  001416                          BEQ     12$             ;;IF NOT:  BR
12533  063400  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
12534  063404  001413                          BEQ     12$             ;;IF NOT:  BR
12535  063406  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
12536  063412  001375                          BNE     11$             ;;IF NOT:  WAIT
12537  063414  017637  000004  001221          MOV     @4(SP),$FATAL   ;;GET ERROR #
12538  063422  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12539  063430  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
12540  063434  105037  063460          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
12541  063440  105037  063457                  CLRB    $LFLG           ;;CLEAR LOG FLAG
12542  063444  105037  063456                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
12543  063450  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12544  063452  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12545  063454  000207                          RTS     PC              ;;RETURN
12546  063456     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
12547  063457     000          $LFLG:  .BYTE   0               ;;LOG FLAG
12548  063460     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
12549          063462                  .EVEN
12550          000200          APTSIZE=200
12551          000001          APTENV=001
12552          000100          APTSPOOL=100
```

```
12553           000040                  APTCSUP=040
12554
12555                                   .NLIST  BEX
```

```
                                    .SBTTL  CONSOLE MESSAGES

063462                              SCTMSG:
063462  005015  040503  047116      .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR>'LF>
063544  051124  041501  020113      .ASCIZ  @TRACK FOR THIS DEVICE@
063572     051                      CLSPRN: .ASCII  @)@
063573     075     000              EQUALS: .ASCIZ  @=@
063575     015  025012     000      PROMPT: .ASCIZ  <CR><LF>@*@
063601     077     000              QSTMRK: .ASCIZ  @?@
063603                              HELPQST:
063603     015  006412  052012      .ASCII <CR><LF><CR><LF>/THIS PROGRAM SHOULD BE HALTED BY TYPE ^C./
063660  005015  044124  020105      .ASCII <CR><LF>/THE PROGRAM WILL BE HALTED AT END OF PASS/<CR><LF><CR>
063736  005015  054524  042520      .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
063772                              UBUSQST:
063772  005015  044103  047101      .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@
064071     015  052412  042523      CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
064130  005015  046522  031460      CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
064155     015  042412  052116      CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
064204  005015  042101  051104              .ASCII  <CR><LF>@ADDRESS MUST BE >160000@
064236  005015  046522  031460      CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
064266  005015  047105  051124      CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
064312  005015  042101  051104              .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
064342  005015  046522  031460      CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
064376  005015  047105  051124      CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
064423                              CNSL07:
064423     015  052012  050131      .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
064501     040  052516  041115      .ASCII  @ NUMBER(S)@
064513     015  052012  051105      .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
064562  005015  044103  047101      XDPMG:  .ASCII  <CR><LF>/CHANGE XXDP PACK,CLEAR LOC 40/
064621     015  051012  051505              .ASCIZ  <CR><LF>/RESTART THE PROGRAM/
064647     015  047012  052117      NOTEX:  .ASCIZ  <CR><LF>/NOT EXIST DRIVE  /

            06-674                  .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.

064674                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
064674  020000          .WORD   OPI                                     ;NOP
064676  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
064700  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
064702  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
064704  020000          .WORD   OPI                                     ;DRIVE CLEAR
064706  030000          .WORD   OPI!IVC                                 ;RELEASE
064710  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
064712  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
064714  020000          .WORD   OPI                                     ;READ IN PRESET
064716  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
064720  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
064722  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
064724  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
064726  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
064730  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
064732  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
064734  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
064736  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
064740  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
064742  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
064744  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
064746  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
064750  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
064752  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
064754  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
064756  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
064760  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
064762  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
064764  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
064766  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
064770  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
064772  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

064774  001             ATNTBL: .BYTE   1.
064775  002                     .BYTE   2.
064776  004                     .BYTE   4.
064777  010                     .BYTE   8.
065000  020                     .BYTE   16.
065001  040                     .BYTE   32.
065002  100                     .BYTE   64.
065003  200                     .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

065004                  RGDTPT:
065004                  MIXED:
065004  000000                  .WORD   0.
065006  000001                  .WORD   1.
065010  000003                  .WORD   3.
065012  000007                  .WORD   7.
065014  000017                  .WORD   15.
065016  000037                  .WORD   31.
065020  000077                  .WORD   63.
065022  000177                  .WORD   127.
065024  000377                  .WORD   255.
065026  000777                  .WORD   511.
065030  001777                  .WORD   1023.
065032  003777                  .WORD   2047.
065034  007777                  .WORD   4095.
065036  017777                  .WORD   8191.
065040  037777                  .WORD   16383.
065042  077777                  .WORD   32767.
065044  177777          ONES:   .WORD   65535.
065046  177777                  .WORD   65535.
065050  077777                  .WORD   32767.
065052  037777                  .WORD   16383.
065054  017777                  .WORD   8191.
065056  007777                  .WORD   4095.
065060  003777                  .WORD   2047.
065062  001777                  .WORD   1023.
065064  000777                  .WORD   511.
065066  000377                  .WORD   255.
065070  000177                  .WORD   127.
065072  000077                  .WORD   63.
065074  000037                  .WORD   31.
065076  000017                  .WORD   15.
065100  000007                  .WORD   7.
065102  000003                  .WORD   3.
065104  000001                  .WORD   1.
065106  000000          ZEROS:  .WORD   0.
065110  000000                  .WORD   0.
065112  000001                  .WORD   1.
065114  000002                  .WORD   2.
065116  000004                  .WORD   4.
065120  000010                  .WORD   8.
065122  000020                  .WORD   16.
065124  000040                  .WORD   32.
065126  000100                  .WORD   64.
065130  000200                  .WORD   128.
065132  000400                  .WORD   256.
065134  001000                  .WORD   512.
065136  002000                  .WORD   1024.
065140  004000                  .WORD   2048.
065142  010000                  .WORD   4096.
065144  020000                  .WORD   8192.
065146  040000                  .WORD   16384.
065150  100000                  .WORD   32768.
065152  100000                  .WORD   32768.
```

```
065154  040000                          .WORD   16384.
065156  020000                          .WORD   8192.
065160  010000                          .WORD   4096.
065162  004000                          .WORD   2048.
065164  002000                          .WORD   1024.
065166  001000                          .WORD   512.
065170  000400                          .WORD   256.
065172  000200                          .WORD   128.
065174  000100                          .WORD   64.
065176  000040                          .WORD   32.
065200  000020                          .WORD   16.
065202  000010                          .WORD   8.
065204  000004                          .WORD   4.
065206  000002                          .WORD   2.
065210  000001                          .WORD   1.
065212  000000                          .WORD   0.
065214  177777                          .WORD   65535.
065216  177776                          .WORD   65534.
065220  177774                          .WORD   65532.
065222  177770                          .WORD   65528.
065224  177760                          .WORD   65520.
065226  177740                          .WORD   65504.
065230  177700                          .WORD   65472.
065232  177600                          .WORD   65408.
065234  177400                          .WORD   65280.
065236  177000                          .WORD   65024.
065240  176000                          .WORD   64512.
065242  174000                          .WORD   63488.
065244  170000                          .WORD   61440.
065246  160000                          .WORD   57344.
065250  140000                          .WORD   49152.
065252  100000                          .WORD   32768.
065254  000000                          .WORD   0.
065256  000000                          .WORD   0.
065260  100000                          .WORD   32768.
065262  140000                          .WORD   49152.
065264  160000                          .WORD   57344.
065266  170000                          .WORD   61440.
065270  174000                          .WORD   63488.
065272  176000                          .WORD   64512.
065274  177000                          .WORD   65024.
065276  177400                          .WORD   65280.
065300  177600                          .WORD   65408.
065302  177700                          .WORD   65472.
065304  177740                          .WORD   65504.
065306  177760                          .WORD   65520.
065310  177770                          .WORD   65528.
065312  177774                          .WORD   65532.
065314  177776                          .WORD   65534.
065316  177777                          .WORD   65535.
065320  125252                  EARLY:  .WORD   43690.
065322  152525                          .WORD   43690./2
065324  125252                          .WORD   43690.
065326  177777                          .WORD   65535.
065330  177776                          .WORD   65534.
065332  177775                          .WORD   65533.
```

```
065334  177773                          .WORD   65531.
065336  177767                          .WORD   65527.
065340  177757                          .WORD   65519.
065342  177737                          .WORD   65503.
065344  177677                          .WORD   65471.
065346  177577                          .WORD   65407.
065350  177377                          .WORD   65279.
065352  176777                          .WORD   65023.
065354  175777                          .WORD   64511.
065356  173777                          .WORD   63487.
065360  167777                          .WORD   61439.
065362  157777                          .WORD   57343.
065364  137777                          .WORD   49151.
065366  077777                          .WORD   32767.
065370  077777                          .WORD   32767.
065372  137777                          .WORD   49151.
065374  157777                          .WORD   57343.
065376  167777                          .WORD   61439.
065400  173777                          .WORD   63487.
065402  175777                          .WORD   64511.
065404  176777                          .WORD   65023.
065406  177377                          .WORD   65279.
065410  177577                          .WORD   65407.
065412  177677                          .WORD   65471.
065414  177737                          .WORD   65503.
065416  177757                          .WORD   65519.
065420  177767                          .WORD   65527.
065422  177773                          .WORD   65531.
065424  177775                          .WORD   65533.
065426  177776                          .WORD   65534.
065430  177777                          .WORD   65535.
065432                          ENRGDT:
```

```
                                .SBTTL  ERROR MESSAGE TABLE

065432  072026  000000          EMT1:   .WORD   EMS1,0
065436  072075  072112  000000  EMT2:   .WORD   EMS2,EMS3,0
065444  072075  072155  000000  EMT3:   .WORD   EMS2,EMS4,0
065452  072220  072250  000000  EMT4:   .WORD   EMS5,EMS6,0
065460  072220  072362  000000  EMT5:   .WORD   EMS5,EMS10,0
065466  077055  074243  000000  EMT6:   .WORD   EMS167,EMS64,0
065474  075011  077102  000000  EMT7:   .WORD   EMS110,EMS170,0
065502  072315  000000          EMT10:  .WORD   EMS7,0
065506  072362  000000          EMT11:  .WORD   EMS10,0
065512  072424  072435  000000  EMT12:  .WORD   EMS11,EMS12,0
065520  072476  072507  072520  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
065532  072572  074243  000000  EMT14:  .WORD   EMS17,EMS64,0
065540  072424  072655  000000  EMT15:  .WORD   EMS11,EMS21,0
065546  072424  072700  073027  EMT16:  .WORD   EMS11,EMS22,EMS27,0
065556  072424  072714  073040  EMT17:  .WORD   EMS11,EMS23,EMS30,0
065566  072424  072742  073040  EMT20:  .WORD   EMS11,EMS24,EMS30,0
065576  072424  072771  073027  EMT21:  .WORD   EMS11,EMS25,EMS27,0
065606  072424  073006  073027  EMT22:  .WORD   EMS11,EMS26,EMS27,0
065616  072424  073050  073040  EMT23:  .WORD   EMS11,EMS31,EMS30,0
065626  072424  073077  073040  EMT24:  .WORD   EMS11,EMS32,EMS30,0
065636  072424  073126  073040  EMT25:  .WORD   EMS11,EMS33,EMS30,0
065646  072424  073154  073040  EMT26:  .WORD   EMS11,EMS34,EMS30,0
065656  072424  073225  073040  EMT27:  .WORD   EMS11,EMS35,EMS30,0
065666  072424  073254  073040  EMT30:  .WORD   EMS11,EMS36,EMS30,0
065676  072424  073303  073040  EMT31:  .WORD   EMS11,EMS37,EMS30,0
065706  072424  073331  073040  EMT32:  .WORD   EMS11,EMS40,EMS30,0
065716  072424  073360  073040  EMT33:  .WORD   EMS11,EMS41,EMS30,0
065726  072424  073406  073040  EMT34:  .WORD   EMS11,EMS42,EMS30,0
065736  072424  073435  073040  EMT35:  .WORD   EMS11,EMS43,EMS30,0
065746  072424  073464  073040  EMT36:  .WORD   EMS11,EMS44,EMS30,0
065756  072424  073537  073040  EMT37:  .WORD   EMS11,EMS45,EMS30,0
065766  074327  072632  000000  EMT40:  .WORD   EMS66,EMS20,0
065774  074515  076223  074523  EMT41:  .WORD   EMS75,EMS141,EMS76,0
066004  076314  076324  074451  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
066016  073631  073747  074523  EMT43:  .WORD   EMS47,EMS53,EMS76,0
066026  074554  073747  074523  EMT44:  .WORD   EMS77,EMS53,EMS76,0
066036  074602  073747  074523  EMT45:  .WORD   EMS100,EMS53,EMS76,0
066046  074630  073747  074523  EMT46:  .WORD   EMS101,EMS53,EMS76,0
066056  074261  074243  000000  EMT47:  .WORD   EMS65,EMS64,0
066064  072476  072520  074215  EMT50:  .WORD   EMS13,EMS15,EMS63,0
066074  072424  073602  073040  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
066106  073631  073747  074377  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
066124  073631  073747  074377  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
066142  073660  073747  074377  EMT54:  .WORD   EMS50,EMS53,EMS67,0
066152  076251  076266  073747  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
066164  073707  074451  074377  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
066202  077055  074461  074377  EMT57:  .WORD   EMS167,EMS73,EMS67,0
066212  074032  074377  075211  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
066230  074436  074032  074377  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
066250  076202  074377  075211  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
066264  000000                  EMT63:  .WORD
066266  076407  076452  074424  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
066306  073735  077502  077663  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
066326  077014  074705  074451  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
066340  077014  074705  074451  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
066352  073602  072632  076707  EMT70:  .WORD   EMS46,EMS20,EMS163,0
066362  074436  074630  076707  EMT71:  .WORD   EMS71,EMS101,EMS163,0
066372  073631  074451  076707  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
066410  073631  074451  076707  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
066426  074032  073747  076707  EMT74:  .WORD   EMS56,EMS53,EMS163,0
066436  074436  074032  076707  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
066456  073660  073747  076707  EMT76:  .WORD   EMS50,EMS53,EMS163,0
066466  076251  076266  073747  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
066500  074515  076223  076707  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
066516  074515  076407  076707  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
066534  077055  074461  076707  EMT102: .WORD   EMS167,EMS72,EMS163,0
066544  076372  076426  074476  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
066556  073707  074476  076707  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
066574  077014  074002  076707  EMT105: .WORD   EMS166,EMS100,EMS163,0
066604  077014  074554  076707  EMT106: .WORD   EMS166,EMS77,EMS163,0
066614  076314  076324  073747  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
066634  075011  075051  075216  EMT110: .WORD   EMS110,EMS116,EMS111,0
066646  075135  072155  000000  EMT111: .WORD   EMS113,EMS54,0
066654  075135  072112  000000  EMT112: .WORD   EMS113,EMS53,0
066662  075011  075211  075216  EMT113: .WORD   EMS110,EMS116,EMS117,EMS114,0
066676  075135  075270  000000  EMT114: .WORD   EMS113,EMS120,0
066704  075305  075745  075771  EMT115: .WORD   EMS121,EMS132,EMS133,0
066714  075350  075745  075771  EMT116: .WORD   EMS122,EMS132,EMS133,0
066724  075405  075745  075771  EMT117: .WORD   EMS123,EMS132,EMS133,0
066734  075450  075745  075771  EMT120: .WORD   EMS124,EMS132,EMS133,
066744  075502  075745  075771  EMT121: .WORD   EMS125,EMS132,EMS133,0
066754  075545  075745  075771  EMT122: .WORD   EMS126,EMS132,EMS133,0
066764  076145  075745  075771  EMT123: .WORD   EMS127,EMS132,EMS133,0
066774  075647  075745  075771  EMT124: .WORD   EMS130,EMS132,EMS133,0
067004  075705  075745  075771  EMT125: .WORD   EMS131,EMS132,EMS133,0
067014  075305  075745  076014  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
067026  075350  075745  076014  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
067040  075405  075745  076014  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
067052  075450  075745  076014  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
067064  075502  075745  076014  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
067076  075545  075745  076014  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
067110  076145  075745  076014  EMT134: .WORD   EMS127,EMS132,EMS134,EMS123,0
067122  075647  075745  076014  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
067134  075705  075745  076014  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
067146  075305  075745  076056  EMT137: .WORD   EMS121,EMS132,EMS135,0
067156  075405  075745  076056  EMT140: .WORD   EMS123,EMS132,EMS135,0
067166  075305  075745  076120  EMT141: .WORD   EMS121,EMS132,EMS136,0
067176  076145  075745  076120  EMT142: .WORD   EMS127,EMS132,EMS136,0
067206  075450  075745  076120  EMT143: .WORD   EMS124,EMS132,EMS136,0
067216  075502  075745  076120  EMT144: .WORD   EMS125,EMS132,EMS136,0
067226  075545  075745  076120  EMT145: .WORD   EMS126,EMS132,EMS136,0
067236  075705  075745  076120  EMT146: .WORD   EMS131,EMS132,EMS136,0
067246  076654  075745  076120  EMT147: .WORD   EMS162,EMS132,EMS136,0
067256  075647  075745  076120  EMT150: .WORD   EMS130,EMS132,EMS136,0
067266  076202  075211  076223  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
067300  076251  075211  076266  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
067312  076314  076324  076353  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
067330  076314  076324  072632  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
067346  076407  076452  076520  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
067360  076372  076426  076520  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
067372  076372  076426  076554  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
067404  076624  076554  076607  EMT160: .WORD   EMS161,EMS156,EMS160,0
067414  072771  073027  076554  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
067426  073006  073027  076554  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
067440  073631  076707  076202  EMT163: .WORD   EMS47,EMS163,EMS140,0
067450  073631  072632  000000  EMT164: .WORD   EMS47,EMS20,0
067456  074032  072632  000000  EMT165: .WORD   EMS56,EMS20,0
067464  073602  072632  000000  EMT166: .WORD   EMS46,EMS20,0
067472  100203  072632  000000  EMT167: .WORD   EMS224,EMS20,0
067500  077055  076520  077030  EMT170: .WORD   EMS167,EMS154,EMS166,0
067510  077055  076520  076572  EMT171: .WORD   EMS167,EMS154,EMS157,0
067520  077055  076520  076574  EMT172: .WORD   EMS167,EMS154,EMS155,0
067530  077131  075746  076771  EMT173: .WORD   EMS171,EMS132,EMS133,0
067540  077131  075746  076014  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
067552  075135  077304  000000  EMT175: .WORD   EMS113,EMS177,0
067560  077326  077343  000000  EMT176: .WORD   EMS200,EMS201,0
067566  077441  071227  076772  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
067600  077441  073707  076772  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
067612  077441  077461  076772  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
067624  077441  073660  073040  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
067636  077441  076266  073040  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
067650  076407  074476  077411  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
067666  073660  074705  074461  EMT205: .WORD   EMS50,EMS103,EMS72,0
067676  074714  074461  077411  EMT206: .WORD   EMS104,EMS73,EMS202,0
067706  074515  076223  075211  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
067720  074515  076407  000000  EMT210: .WORD   EMS75,EMS150,0
067726  073707  074451  075211  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
067742  076251  073747  075211  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
067756  073660  074705  073747  EMT213: .WORD   EMS50,EMS103,EMS53,0
067766  073735  077502  077030  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
070006  073735  075246  074032  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
070020  074032  073040  074243  EMT216: .WORD   EMS56,EMS30,EMS64,0
070030  073602  073040  074243  EMT217: .WORD   EMS46,EMS30,EMS64,0
070040  073050  073040  074243  EMT220: .WORD   EMS31,EMS30,EMS64,0
070050  073631  073040  074243  EMT221: .WORD   EMS47,EMS30,EMS64,0
070060  073735  075246  074554  EMT222: .WORD   EMS52,EMS117,EMS77,0
070070  073735  075246  074215  EMT223: .WORD   EMS52,EMS117,EMS63,0
070100  000000                  EMT224: .WORD
070102  000000                  EMT225: .WORD
070104  000000                  EMT226: .WORD
070106  000000                  EMT227: .WORD
070110  000000                  EMT230: .WORD
070112  000000                  EMT231: .WORD
070114  000000                  EMT232: .WORD
070116  000000                  EMT233: .WORD
070120  000000                  EMT234: .WORD
070122  000000                  EMT235: .WORD
070124  000000                  EMT236: .WORD
070126  000000                  EMT237: .WORD
070130  000000                  EMT240: .WORD
070132  000000                  EMT241: .WORD
070134  000000                  EMT242: .WORD
070136  000000                  EMT243: .WORD
070140  000000                  EMT244: .WORD
070142  000000                  EMT245: .WORD
070144  077055  075745  077541  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
070154  077055  075745  077566  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
070166  077055  077577  077566  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
070202  077615  077640  000000  EMT251: .WORD   EMS212,EMS213,0
070210  077014  077615  000000  EMT252: .WORD   EMS165,EMS212,0
070216  076372  076426  076554  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
070234  077441  074630  073040  EMT254: .WORD   EMS203,EMS101,EMS30,0
070244  077441  077055  073040  EMT255: .WORD   EMS203,EMS167,EMS30,0
070254  077441  074602  073040  EMT256: .WORD   EMS203,EMS100,EMS30,0
070264  072424  073602  073040  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
070276  073735  075346  074032  EMT260: .WORD   EMS62,EMS117,EMS56,EMS102,0
070310  073735  077502  100006  EMT261: .WORD   EMS62,EMS206,EMS220,EMS206,EMS115,EMS51,EMS72,0
070330  074714  074461  077411  EMT262: .WORD   EMS104,EMS73,EMS202,0
070340  073660  073747  074[illegible]  EMT263: .WORD   EMS50,EMS53,EMS102,0
070350  074032  073747  074[illegible]  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
070370  074436  074032  074[illegible]  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
070410  076251  076266  073747  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
070422  073660  076353  074[illegible]  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
070440  073631  073747  074[illegible]  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
070454  073631  073747  074[illegible]  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
070472  074515  076223  074[illegible]  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
070510  073707  074476  074[illegible]  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
070526  074215  073747  074107  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
070544  075216  073747  073360  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
070556  073631  073747  074107  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
070572  073631  073747  074107  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
070610  074032  073747  074107  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
070630  074436  074032  073747  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
070652  077014  074602  074705  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
070664  077014  074630  074705  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
070676  077014  074554  074705  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
070710  073602  073040  074243  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
070722  073660  073747  074107  EMT306: .WORD   EMS50,EMS53,EMS57,0
070732  073660  076353  075211  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
070752  076251  076266  073747  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
070764  074714  074705  073747  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
070776  074743  074705  073747  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
071010  076314  076324  074705  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
071030  073406  074705  073747  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
071042  073050  074705  073747  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
071054  074436  073050  074705  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
071066  073435  074705  074107  EMT317: .WORD   EMS43,EMS103,EMS57,0
071076  073537  074705  074107  EMT320: .WORD   EMS45,EMS103,EMS57,0
071106  073464  074705  074107  EMT321: .WORD   EMS44,EMS103,EMS57,0
071116  074772  072632  000000  EMT322: .WORD   EMS106,EMS20,0
071124  073254  074705  074107  EMT323: .WORD   EMS36,EMS103,EMS57,0
071134  077204  073254  074705  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
071146  077164  073254  074705  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
071160  072476  077221  072520  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
071176  076372  076426  074476  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
071210  074327  073747  077227  EMT330: .WORD   EMS66,EMS53,EMS175,0
071220  073126  074705  073747  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
071232  073331  074705  073747  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
071244  073707  074476  074107  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
071262  074515  076223  074107  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
071300  074515  076407  076452  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
071320  074135  074150  074177  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
071330  075216  075246  073154  EMT337: .WORD   EMS116,EMS117,EMS34,0
071340  073154  073747  073761  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
071352  074003  073154  000000  EMT341: .WORD   EMS55,EMS34,0
071360  073735  075246  074032  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
071372  073735  075246  073537  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
071404  073735  075246  073464  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
071416  073735  075246  100025  EMT345: .WORD   EMS52,EMS117,EMS221,0
071426  074772  072632  075211  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
071442  073735  077502  100075  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
071454  074515  076407  074656  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
071472  077055  074461  074656  EMT351: .WORD   EMS167,EMS73,EMS102,0
071502  077701  000000          EMT352: .WORD   EMS215,0
071506  077752  077441  077722  EMT353: .WORD   EMS217,EMS203,EMS216,0
071516  100025  072632  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
071524  100255  101071  101150  EHT1:    .WORD   EH1,STSH1,STSH2,STSH4,0
071536  101071  101150  101301  EHT2:    .WORD   STSH1,STSH2,STSH4,0

071546  100274  000000          EHT110:  .WORD   EH110,0
071552  100303  000000          EHT111:  .WORD   EH111,0

071556  100322  000000          EHT114:  .WORD   EH114,0
071562  100351  101071  101150  EHT223:  .WORD   EH223,STSH1,STSH2,STSH4,0
071574  100377  101071  101150  EHT256:  .WORD   EH256,STSH1,STSH2,STSH4,0

071606  100454  101071  101150  EHT336:  .WORD   EH336,STSH1,STSH2,STSH4,0
071620  100513  101071  101150  EHT337:  .WORD   EH337,STSH1,STSH2,STSH4,0
071632  100652  101071  101150  EHT344:  .WORD   EH344,STSH1,STSH2,STSH4,0

071644  101012  000000          EHT353:  .WORD   EH353,0
```

```
071650  101342  101436  101454  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
071660  101436  101454  101506  EDT2:    .WORD   STSD1,STSD2,STSD4

071666  101350                  EDT110:  .WORD   ED110
071670  101354                  EDT111:  .WORD   ED111

071672  101362                  EDT114:  .WORD   ED114
071674  101372  101436  101454  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4

071704  101402  101436  101454  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
071714  101414  101436  101454  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
071724  101414  101436  101454  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0

071736  101426                  EDT353:  .WORD   ED353
```

```
071740  101521  101537  101537  EFT1:    .WORD   EF111,STSF,STSF,STSF
071750  101537  101537  101537  EFT2:    .WORD   STSF,STSF,STSF

071756  101520                  EFT110:  .WORD   EF110
071760  101521                  EFT111:  .WORD   EF111

071762  101523                  EFT114:  .WORD   EF114
071764  101523  101537  101537  EFT223:  .WORD   EF114,STSF,STSF,STSF

071774  101526  101537  101537  EFT336:  .WORD   EF336,STSF,STSF,STSF
072004  101526  101537  101537  EFT337:  .WORD   EF336,STSF,STSF,STSF
072014  101526  101537  101537  EFT344:  .WORD   EF336,STSF,STSF,STSF

072024  101523                  EFT353:  .WORD   EF114
```

```
                                .SBTTL  ERROR MESSAGE STRINGS

072026 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
072075    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
072112 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
072155    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
072220 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED @
072250 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
072315    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
072362 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
072424 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
072435    106 047124 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
072476 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
072507    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
072520 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
072542 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
072572 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
072632 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
072655    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
072700 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
072714 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
072742 040442 042517 020042 EMS24:  .ASCIZ  @"ADE" (RMER1, BIT 09) @
072771    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
073006 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
073027    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
073040 052123 052101 051523 EMS30:  .ASCIZ  @STATUS @
073050 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
073077    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
073126 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
073154 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
073225    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
073254 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
073303    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
073331    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
073360 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
073406 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
073435    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
073464 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
073537    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
073602 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
073631    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
073660 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
073707    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
073735    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
073747    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
073761    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
074003    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
074032 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
074107    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
074135    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
074150 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
074177    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
074215    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
074243    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
074261    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
074327    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
074377     104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
074424  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
074436  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
074451     111  020123  042523  EMS72:  .ASCIZ  @IS SET @
074461     104  042111  047040  EMS73:  .ASCIZ  @OID NOT SET @
074476  044504  020104  047516  EMS74:  .ASCIZ  @OID NOT RESET @
074515     114  051517  020124  EMS75:  .ASCIZ  @LOST @
074523     104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
074554  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
074602  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
074630  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
074656  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
074705     105  051122  051117  EMS103: .ASCIZ  @ERROR @
074714  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
074743     042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
074772  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
075011     102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
075040  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
075051     127  042510  020116  EMS112: .ASCIZ  @WHEN READING/WRITING RH REGISTERS @
075113     101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
075135     124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
075165     116  047117  051105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
075211     040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
075216  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
075246  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
075270  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
075305     103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
075350  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
075405     103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
075450  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
075502  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
075545     115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
075610  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
075647     105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
075705     115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
075745     116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
075771     125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
076014  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E. BIT 5 OF @
076056  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
076120  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
076145     104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
076202  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
076223     042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
076251     104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
076266  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
076314  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
076324  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
076353     123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
076372  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
076407     040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
076426  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
076452  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
076476  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
076520  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
076534  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
076554  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
076572 052122 020103 047503 EMS157: .ASCIZ  @RTC COMMAND @
076607    122 050111 041440 EMS160: .ASCIZ  @RIP C    ND @
076624 043117 051506 052105 EMS161: .ASCIZ  @OFFSE   STER (RMOF) @
076654 051105 047522 020122 EMS162: .ASCIZ  @ RROR R   ER #2, RMER2, @
076707    104 051126 047111 EMS163: .ASCIZ  @DURING     BRATE @
076733    111 020123 047111 EMS164: .ASCIZ  @IS INTERMITTE   OR DRIVE DIDNT DROP ON @
077002 054503 044514 042116       .ASCIZ  @CYLINDER @
077014 047125 054105 042520 EMS165: .ASCIZ  @ INEXPECTED @
077030 042522 040503 044514 EMS166: .ASCIZ  @RECALIBRATE COMMAND @
077055    042 052101 021101 EMS167: .ASCIZ  @"ATA" (RMOS, BIT15) @
077102 044127 047105 051040 EMS170: .ASCIZ  @WHEN RE   ING REGISTER @
077131    105 051122 051117 EMS171: .ASCIZ  @ ROR REGISTER #2, RMER2, @
077164 047516 051116 041505 EMS172: .ASCIZ  @UNRECOVERABLE @
077204 042522 047503 042526 EMS173: .ASCIZ  @ ECOVERABLE @
077221    104 052101 020101 EMS174: .ASCIZ  @DATA @
077227    104 051125 047111 EMS175: .ASCIZ  @DURING WRITE COMMAND @
077255    042 050117 021105 EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
077304 047111 053440 044522 EMS177: .ASCIZ  @IN WRITE PROTECT @
077326 040503 020116 047516 EMS200: .ASCIZ  @CAN NOT SET @
077343    104 040511 047107 EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
077411    104 051125 047111 EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
077441    111 041516 051117 EMS203: .ASCIZ  @INCORRECT @
077454 053442 046122 020042 EMS204: .ASCIZ  @"WRL" (RMOS, BIT 11) @
077502 054105 041505 052125 EMS205: .ASCIZ  @EXECUTED @
077514 044527 044124 041440 EMS206: .ASCIZ  @WITH COMP ERROR SET @
077541    042 047507 020042 EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
077566 051127 052111 047111 EMS210: .ASCIZ  @WRITING @
077577    127 051501 051040 EMS211: .ASCIZ  @WAS RESET BY @
077615    120 047522 051107 EMS212: .ASCIZ  @PROGRAM INTERRUPT @
077640 040527 020123 047516 EMS213: .ASCIZ  @WAS NOT GENERATED @
077663    123 042505 020113 EMS214: .ASCIZ  @SEEK COMMAND @
077701    120 047522 051107 EMS215: .ASCIZ  @PROGRAM TIMEOUT @
077722 052504 044522 043516 EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
077752 047514 045517 040440 EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
100005    123 040505 041522 EMS220: .ASCIZ  @SEARCH COMMAND @
100025    102 042101 051440 EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
100075    101 042040 052101 EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
100126 042510 042101 051105 EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
100203    116 047117 051105 EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
100255    105  050130  052103  EH1:    .ASCIZ  @EXPCTD  RECEVD@
100274  052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
100303    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

100322  042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
100351    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
100377    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
100426  052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

100454  042107  042101  051522  EH336:  .ASCIZ  @GDDADRS  GDDATA  BDADRS  BDDATA@
100513    122  041515  031123  EH337:  .ASCII  @RMCS2    STATUS  FAILING   DATA@<CR><LF>
100553    137  057537  057537          .ASCII  @+++++++++++++++  ++++++++++++++@<CR><LF>
100613    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

100652  046522  051105  020061  EH344:  .ASCII  @RMER1    STATUS  HEADER  FAILING@<CR><LF>
100713    137  057537  057537          .ASCII  @+++++++++++++++   WORD     BIT@<CR><LF>
100752  054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
101012  054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
101032  041474  037122  046074          .ASCIZ  @<CR><LF> RMLA     RMLA     RMOF @

101071    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
101136  020040  051040  040515          .ASCIZ  @   RMAS@<CR><LF>
101150  051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
101215    040  020040  051040          .ASCIZ  @    RMEC1   RMEC2@<CR><LF>
101241    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@<CR><LF>
101301    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@<CR><LF>
```

```
                101342
                                                .EVEN
101342  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
101350  001276  000000          ED110:  .WORD   $BASE,0
101354  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

101362  001356  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
101372  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

101402  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

101414  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
101426  001140  001142  001432  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

101436  001330  001340  001342  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
101454  001332  001334  001336  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
101470  001376  000000          .WORD   RMEC2I,0
101474  001336  001364  001362  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
101506  001354  001370  001356  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
101520  000             EF110:  .BYTE   0
101521  000     000     EF111:  .BYTE   0,0
101523  000     000     000     EF114:  .BYTE   0,0,0
101526  000     000     000     EF336:  .BYTE   0,0,0,0
101532  000     000     000     EF337:  .BYTE   0,0,0,0,0

101537  000     000     000     STSF:   .BYTE   0,0,0,0,0,0,0
```

```
                        .EVEN
101546  000402  MFGFIL: .BLKW   258.
102552  177777          .WORD   -1
102554  177777          .WORD   -1
102556  000402  USRFIL:         .BLKW   258.
103562  177777          .WORD   -1
103564  177777          .WORD   -1
                        .EVEN
```

```
103566                          BUFFER:
103566  000402                  BUFONE: .BLKW   258.
104572  000402                  BUFTWO: .BLKW   258.

        103566                  .       =       BUFFER
103566                          HELP:
103566  005015                  .ASCII  <CR><LF>
                                ;.ASCII @CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE @<CR><LF>
                                ;.ASCII @BAD HEADERS ON THE PACK SO BE SURE TO FORMAT THE PACKS@<CR><LF>
                                ;.ASCII @WHEN DONE.@<CR><LF>
103570  046011  051511  020124  .ASCII  @       LIST OF TESTS@<CR><LF>
103610  057411  057537  057537  .ASCII  @       **********@<CR><LF>
103630  030524  041411  047117  .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
103663     124  004462  042504  .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
103715     124  004463  051104  .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
103741     124  004464  047506  .ASCII  @T4     FORMAT ZEROS - 18@<CR><LF>
103767     124  004465  047506  .ASCII  @T5     FORMAT ZEROS - 16@<CR><LF>
104015     124  004466  042532  .ASCII  @T6     ZERO FILL TEST@<CR><LF>
104040  033524  043011  051117  .ASCII  @T7     FORMAT CHECK ZEROS - 16@<CR><LF>
104074  030524  004460  047506  .ASCII  @T10    FORMAT CHECK ZEROS W/ WCE ERROR@<CR><LF>
104141     124  030461  043011  .ASCII  @T11    FORMAT ONES - 16@<CR><LF>
104167     124  031061  043011  .ASCII  @T12    FORMAT CHECK ONES - 16@<CR><LF>
104223     124  031461  043011  .ASCII  @T13    FORMAT CHECK ONES W/ WCE ERRORS@<CR><LF>
104270  030524  004464  047506  .ASCII  @T14    FORMAT MULTIPLE SECTORS@<CR><LF>
104325     124  032461  043011  .ASCII  @T15    FORMAT W/ HEAD SWITCHING@<CR><LF>
104363     124  033061  043011  .ASCII  @T16    FORMAT W/ MID TRANSFER SEEK@<CR><LF>
104424  030524  004467  047506  .ASCII  @T17    FORMAT W/ IMPLIED SEEK@<CR><LF>
104460  031124  004460  047506  .ASCII  @T20    FORMAT EACH SECTOR ADDRESS@<CR><LF>
104520  031124  004461  047506  .ASCII  @T21    FORMAT EACH TRACK ADDRESS@<CR><LF>
104557     124  031062  043011  .ASCII  @T22    FORMAT PRIME CYLINDERS@<CR><LF>
104613     124  031462  043011  .ASCII  @T23    FORMAT LAST SECTOR@<CR><LF>
104643     124  032062  043011  .ASCII  @T24    FORMAT W/ AOE ERROR@<CR><LF>
104674  031124  004465  047506  .ASCII  @T25    FORMAT INVALID SECTOR ADDRESS@<CR><LF>
104737     124  033062  043011  .ASCII  @T26    FORMAT INVALID TRACK ADDRESS@<CR><LF>
105001     124  033462  043011  .ASCII  @T27    FORMAT INVALID CYLINDER ADDRESS@<CR><LF>
105046  031524  004460  047506  .ASCII  @T30    FORMAT AT OFFSET@<CR><LF>
105074  031524  004461  053111  .ASCII  @T31    IVC FORMAT TEST@<CR><LF>
105121     124  031063  043011  .ASCII  @T32    FORMAT ERROR TEST - 18@<CR><LF>
105155     124  031463  043011  .ASCII  @T33    FORMAT ERROR TEST - 16@<CR><LF>
105211     124  032063  043011  .ASCII  @T34    FORMAT HCE, FIRST HEADER WORD@<CR><LF>
105254  031524  004465  047506  .ASCII  @T35    FORMAT HCE, SECOND HEADER WORD@<CR><LF>
105320  005015                  .ASCII  <CR><LF>
105322  047411  042520  040522  .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
105360  057411  057537  057537  .ASCII  @       **********@<CR><LF>
105416  005015                  .ASCII  <CR><LF>
105420  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
105436  026455  026455  026455  .ASCII  @------                  ----------@<CR><LF>
105474  020040  032461  004411  .ASCII  @   15                   HALT ON ERROR@<CR><LF>
105521     040  030440  004464  .ASCII  @   14                   LOOP ON TEST@<CR><LF>
105545     040  030440  004463  .ASCII  @   13                   INHIBIT ERROR TYPEOUTS@<CR><LF>
105603     040  030440  004462  .ASCII  @   12                   @<CR><LF>
105613     040  030440  004461  .ASCII  @   11                   INHIBIT ITERATIONS@<CR><LF>
105645     040  030440  004460  .ASCII  @   10                   BELL ON ERROR@<CR><LF>
105672  020040  034440  004411  .ASCII  @    9                   LOOP ON ERROR@<CR><LF>
105717     040  020040  004470  .ASCII  @    8                   LOOP ON TEST IN SWR<7:0>@<CR><LF>
```

```
105757     040  020040  004467  .ASCII  @   7          TN128@'CR><LF>
105774  020040  033040  004411  .ASCII  @   6          TN64@<CR><LF>
106010  020040  032440  004411  .ASCII  @   5          TN32@<CR><LF>
106024  020040  032040  004411  .ASCII  @   4          TN16@<CR><LF>
106040  020040  031440  004411  .ASCII  @   3          TN8@<CR>'LF>
106053     040  020040  004462  .ASCII  @   2          TN4@<CR><LF>
106066  020040  030440  004411  .ASCII  @   1          TN2@<CR><LF>
106101     040  020040  004460  .ASCII  @   0          TN1@<CR><LF>
106114  005015                  .ASCII  <CR><LF>
106116  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1252#  1360   1401
ACDW1 = 000000          1360   1403
ACDW2 = 000000          1360   1404
ACKSTS  047656          8083   8271   10326#
ACPUOP= 000000          1360   1375
ADDW0 = 000000          1360   1405
ADDW1 = 000000          1360   1406
ADDW10= 000000          1360
ADDW11= 000000          1360
ADDW12= 000000          1360
ADDW13= 000000          1360
ADDW14= 000000          1360
ADDW15= 000000          1360
ADDW2 = 000000          1360   1407
ADDW3 = 000000          1360   1408
ADDW4 = 000000          1360   1409
ADDW5 = 000000          1360   1410
ADDW6 = 000000          1360   1411
ADDW7 = 000000          1360   1412
ADDW8 = 000000          1360
ADDW9 = 000000          1360
ADEVCT= 000000          1360   1366
ADEVM = 000000          1360   1402
ADR   = 000001          3524#  3525   3566#  3567#  3568   3610#  3611#  3612   3648#  3649#  3650   3816#  3817#
                        3818   3984#  3985#  3986   4154#  4155#  4156   4314#  4315#  4316   4502#  4503#  4504
                        4670#  4671#  4672   4830#  4831#  4832   5018#  5019#  5020   5177#  5178#  5179   5336#
                        5337#  5338   5495#  5496#  5497   5660#  5661#  5662   5805#  5806#  5807   5950#  5951#
                        5952   6095#  6096#  6097   6180#  6181#  6182   6262#  6263#  6264   6409#  6410#  6411
                        6556#  6557#  6558   6702#  6703#  6704   6877#  6878#  6879   7030#  7031#  7032   7168#
                        7169#  7170   7306#  7307#  7308   7542#  7543#  7544
AENV  = 000000          1360   1371
AENVM = 000000          1360   1372
AFATAL= 000000          1360   1363
AMADR1= 000000          1360   1388
AMADR2= 000000          1360   1392
AMADR3= 000000          1360   1395
AMADR4= 000000          1360   1398
AMAMS1= 000000          1360   1382
AMAMS2= 000000          1360   1390
AMAMS3= 000000          1360   1393
AMAMS4= 000000          1360   1396
AMSGAD= 000000          1360   1368
AMSGLG= 000000          1360   1369
AMSGTY= 000000          1360   1362
AMTYP1= 000000          1360   1383
AMTYP2= 000000          1360   1391
AMTYP3= 000000          1360   1394
AMTYP4= 000000          1360   1397
AOE   = 001000          1032#  1043   9496   9511   9526   9541   9545   9792   9796   12555
APASS = 000000          1360   1365
APE   = 100000          1231#
APRIOR= 000000          1360
APTCSU= 000040         11902  12553#
APTENV= 000001         11895  12087  12509  12551#
APTSIZ= 000200          3268  12550#
APTSPD= 000100         11897  12511  12552#
```

```
ARGS   = 000004          3669#  3706#  3738#  3745#  3752#  3781#  3788#  3795#  3804#  3837#  3874#  3906#  3913#
                         3920#  3949#  3956#  3963#  3972#  4005#  4042#  4075#  4082#  4089#  4119#  4126#  4133#
                         4142#  4175#  4212#  4244#  4251#  4258#  4286#  4293#  4300#  4335#  4372#  4404#  4411#
                         4418#  4449#  4460#  4523#  4560#  4592#  4599#  4606#  4635#  4642#  4649#  4658#  4691#
                         4728#  4760#  4767#  4774#  4802#  4809#  4816#  4851#  4888#  4920#  4927#  4934#  4965#
                         4976#  5039#  5076#  5108#  5115#  5122#  5150#  5157#  5164#  5198#  5235#  5267#  5274#
                         5281#  5309#  5316#  5323#  5357#  5394#  5426#  5433#  5440#  5468#  5475#  5482#  5516#
                         5555#  5588#  5595#  5602#  5630#  5637#  5644#  5681#  5719#  5726#  5733#  5762#  5769#
                         5776#  5785#  5826#  5864#  5871#  5878#  5907#  5914#  5921#  5930#  5971#  6009#  6016#
                         6023#  6052#  6059#  6066#  6075#  6104#  6152#  6159#  6166#  6188#  6234#  6241#  6248#
                         6283#  6321#  6328#  6335#  6344#  6375#  6382#  6389#  6430#  6468#  6475#  6482#  6491#
                         6522#  6529#  6536#  6577#  6615#  6622#  6629#  6638#  6669#  6676#  6683#  6723#  6773#
                         6780#  6787#  6835#  6842#  6849#  6858#  6898#  6949#  6963#  6972#  7003#  7017#  7051#
                         7089#  7096#  7103#  7133#  7144#  7155#  7189#  7227#  7234#  7241#  7271#  7282#  7293#
                         7328#  7375#  7382#  7389#  7418#  7433#  7448#  7468#  7486#  7513#  7564#  7611#  7618#
                         7625#  7654#  7665#  7683#  7708#

ASNDA    001500          1484#  8461*  8490   8536*  8537   8539*  8540*  8541   8543*  8588
ASNDC    001476          1483#  8460*  8489   8544*  8545   8547*  8587
ASWREG=  000000          1360   1373
ATA    = 100000          1012#  9323   9324   9326   10058  10059  10094  10097  10589  10590  10629  10632  12555
ATESTN=  000000          1360   1364
ATNMSK=  000377          1049#  10786
ATNTBL   064774          3466   3484   12555#
AUNIT  = 000000          1360   1367
AUSWR  = 000000          1360   1374
AVECT1=  120254          1253#  1360   1399
AVECT2=  000000          1360   1400
A16    = 000400          1202#
A17    = 001000          1201#
BACK   = 000000          3669#  3674   3706#  3710   3738#  3742   3745#  3749   3752#  3756   3781#  3785   3788#
                         3792   3795#  3799   3804#  3810   3837#  3842   3874#  3878   3906#  3910   3913#  3917
                         3920#  3924   3949#  3953   3956#  3960   3963#  3967   3972#  3978   4005#  4010   4042#
                         4046   4075#  4079   4082#  4086   4089#  4093   4119#  4123   4126#  4130   4133#  4137
                         4142#  4148   4175#  4180   4212#  4216   4244#  4248   4251#  4255   4258#  4262   4286#
                         4290   4293#  4297   4300#  4304   4335#  4340   4372#  4376   4404#  4408   4411#  4415
                         4418#  4422   4449#  4453   4460#  4464   4523#  4528   4560#  4564   4592#  4596   4599#
                         4603   4606#  4610   4635#  4639   4642#  4646   4649#  4653   4658#  4664   4691#  4696
                         4728#  4732   4760#  4764   4767#  4771   4774#  4778   4802#  4806   4809#  4813   4816#
                         4820   4851#  4856   4888#  4892   4920#  4924   4927#  4931   4934#  4938   4965#  4969
                         4976#  4980   5039#  5044   5076#  5080   5108#  5112   5115#  5119   5122#  5126   5150#
                         5154   5157#  5161   5164#  5168   5198#  5203   5235#  5239   5267#  5271   5274#  5278
                         5281#  5285   5309#  5313   5316#  5320   5323#  5327   5357#  5362   5394#  5398   5426#
                         5430   5433#  5437   5440#  5444   5468#  5472   5475#  5479   5482#  5486   5516#  5521
                         5555#  5559   5588#  5592   5595#  5599   5602#  5606   5630#  5634   5637#  5641   5644#
                         5648   5681#  5686   5719#  5723   5726#  5730   5733#  5737   5762#  5766   5769#  5773
                         5776#  5780   5785#  5791   5826#  5831   5864#  5868   5871#  5875   5878#  5882   5907#
                         5911   5914#  5918   5921#  5925   5930#  5936   5971#  5976   6009#  6013   6016#  6020
                         6023#  6027   6052#  6056   6059#  6063   6066#  6070   6075#  6081   6104#  6109   6152#
                         6156   6159#  6163   6166#  6170   6188#  6193   6234#  6238   6241#  6245   6248#  6252
                         6283#  6288   6321#  6325   6328#  6332   6335#  6339   6344#  6349   6375#  6379   6382#
                         6386   6389#  6393   6430#  6435   6468#  6472   6475#  6479   6482#  6486   6491#  6496
                         6522#  6526   6529#  6533   6536#  6540   6577#  6582   6615#  6619   6622#  6626   6629#
                         6633   6638#  6643   6669#  6673   6676#  6680   6683#  6687   6723#  6728   6773#  6777
                         6780#  6784   6787#  6791   6835#  6839   6842#  6846   6849#  6853   6858#  6864   6898#
                         6903   6949#  6953   6963#  6967   6972#  6977   7003#  7007   7017#  7021   7051#  7056
                         7089#  7093   7096#  7100   7103#  7107   7133#  7137   7144#  7148   7155#  7159   7189#
```

```
                        5002    5744    5787    5889    5932    6034    6077    6117    6201    6354    6501    6648    6812
                        6860    6982    7115    7253    7400    7459    7478    7636    7675   12555#
CC     = 004000         1128#
CH     = 002000         1129#
CHRCNT  033741           7848*   7866*   7872*   7873    7876*   7880    7885*   7896*   7944#
CKSWR  = 104410         11966   12071   12097   12448#
CLEAN  = ****** U        7462   12555
CLKADR  001502           1485#   8961*   8963    8967*   8969    8994
CLKVCT  001504           1486#   8962*   8968*
CLF    = 000040          1219#   3315   10155
CLRSTS  046776           8032   10191#
CLSPRN  063572           3386    3402    3423   12555#
CMNSTA  006772           3338    3473#
CMPBUF  037064           3804    3972    4142    4658    5785    5930    6075    6858    8690#
CMPERR  057122          11570#
CNSL00  064071           3348   12555#
CNSL01  064130           3383   12555#
CNSL02  064155           3392   12555#
CNSL03  064236           3397   12555#
CNSL04  064266           3408   12555#
CNSL05  064342           3418   12555#
CNSL06  064376           3429   12555#
CNSL07  064423           3442   12555#
CNTCLR  046660           3573    3617    8012   10147#
CONT   = 000100          1092#
CR     = 000015           843#   3456    7864   11652   11941   11951   12555
CRLF   = 000200           844#   3292   11912   11951
CTLFG   001326           1415#   3512*  11643   11657*
CYLMSK= 001777           1119#   8628
DBCK   = 100000          1065#
DBEN   = 040000          1066#
DBL    = 002000          1236#
DCK    = 100000          1026#   1043    8366    8698    9654    9674   11264   11267
DOISP  = 177570           850#   1336    3256
DEBL   = 020000          1067#
DEVSEL  045206           7999    9857#
DISPLA  001156           1336#   3256*   3264*  12022*  12074*
DISPRE  000174           1263#   3264
DLT    = 100000          1208#   9498   11343   11346
DMD    = 000001          1079#   1098#   6907
DPE    = 000010          1151#   8367    8373    9160    9440    9936   10475   10896   11317   11320
DPEHI  = 040000          1232#
DPELO  = 020000          1233#
DPR    = 000400          1019#  10097   10294   10295   10761
DRQ    = 004000          1105#
DRVCLR= 000010            963#   8220   10749
DRVSTS  052214           8238   10740#
DRY    = 000200          1020#   9127    9223    9226    9227   10097   10294   10295   10761
DSWR   = 177570           849#   1335    3255
DTASTS  053016           3745    3788    3913    3956    4082    4126    4251    4293    4411    4460    4599    4642    4767
                         4809    4927    4976    5115    5157    5274    5316    5433    5475    5595    5637    5726    5769
                         5871    5914    6016    6059    6159    6241    6328    6382    6475    6529    6622    6676    6780
                         6842    7096    7144    7234    7282    7382    7433    7448    7468    7618    7665    8341   10871#
DTE    = 010000          1029#   1043   11130   11148   11151
DTO    = 010000          1068#
DULPRT= 024024           1108#   3632    3636    3639
```

```
DVA    = 004000          948#  3321  3592  8850  8923  9088  9091  9882 10200 10748 10749
DVC    = 000200         1150# 10029 10284 10536 10574 10577 10710 11083 11087 11090 11135 11509 11514
                        11517
EARLY    065320         12555#
EBL    = 020000          1085#
ECH    = 000100          1035#  1043  8366  8700  9637  9652  9670  9676 11274 12555
ECI    = 004000          1113#  8702  9672 11271
ECRC   = 001000          1089#
EDT1     071650          1520  1527  1534  1541  1548  1555  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  1772  1779  1786  1793
                         1800  1807  1815  1822  1829  1836  1843  1851  1859  1867  1881  1888  1895
                         1902  1909  1916  1923  1931  1938  1945  1952  1959  1966  1973  1980  1987
                         1994  2001  2008  2015  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  2281  2288  2295
                         2302  2309  2316  2331  2338  2345  2352  2366  2373  2380  2387  2394  2401
                         2408  2415  2422  2429  2436  2443  2450  2457  2464  2471  2478  2485  2492
                         2513  2520  2527  2534  2541  2681  2688  2695  2716  2723  2730  2744  2751
                         2758  2765  2772  2779  2786  2793  2801  2808  2816  2823  2830  2838  2846
                         2854  2863  2871  2879  2886  2893  2900  2907  2914  2921  2928  2935  2942
                         2949  2956  2963  2970  2977  2984  2991  2998  3005  3012  3019  3026  3033
                         3040  3047  3054  3061  3068  3075  3110  3117  3131  3138  3145  3152  3159
                         3180 12555#
EDT110   071666          2022 12555#
EDT111   071670          2029  2036 12555#
EDT114   071672          2050 12555#
EDT2     071660          2499  2506  2702  2709 12555#
EDT223   071674          2548  2737 12555#
EDT336   071704          2324  3082  3096  3103 12555#
EDT337   071714          3089 12555#
EDT344   071724          3124 12555#
EDT353   071736          3173 12555#
ED1      101342         12555#
ED110    101350         12555#
ED111    101354         12555#
ED114    101362         12555#
ED223    101372         12555#
ED336    101402         12555#
ED337    101414         12555#
ED353    101426         12555#
EECC   = 000020          1094#
EFT1     071740          1521  1528  1535  1542  1549  1556  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  1773  1780  1787  1794
                         1801  1808  1816  1823  1830  1837  1844  1852  1860  1868  1882  1889  1896
                         1903  1910  1917  1924  1932  1939  1946  1953  1960  1967  1974  1981  1988
                         1995  2002  2009  2016  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  2282  2289  2296
                         2303  2310  2317  2332  2339  2346  2353  2367  2374  2381  2388  2395  2402
                         2409  2416  2423  2430  2437  2444  2451  2458  2465  2472  2479  2486  2493
                         2514  2521  2528  2535  2542  2682  2689  2696  2717  2724  2731  2745  2752
                         2759  2766  2773  2780  2787  2794  2802  2809  2817  2824  2831  2839  2847
                         2855  2864  2872  2880  2887  2894  2901  2908  2915  2922  2929  2936  2943
                         2950  2957  2964  2971  2978  2985  2992  2999  3006  3013  3020  3027  3034
```

		3041	3048	3055	3062	3069	3076	3111	3118	3132	3139	3146	3153	3160
		3181	12555#											
EFT110	071756	2023	12555#											
EFT111	071760	2030	2037	12555#										
EFT114	071762	2051	12555#											
EFT2	071750	2500	2507	2703	2710	12555#								
EFT223	071764	2549	2738	12555#										
EFT336	071774	2325	3083	3097	3104	12555#								
EFT337	072004	3090	12555#											
EFT344	072014	3125	12555#											
EFT353	072024	3174	12555#											
EF110	101520	12555#												
EF111	101521	12555#												
EF114	101523	12555#												
EF336	101526	12555#												
EF337	101532	12555#												
EHT1	071524	1519	1526	1533	1540	1547	1554	1568	1575	1582	1589	1596	1603	1610
		1617	1624	1631	1638	1645	1652	1659	1666	1673	1680	1687	1694	1701
		1708	1715	1722	1729	1736	1743	1750	1757	1764	1771	1778	1785	1792
		1799	1806	1814	1821	1828	1835	1842	1850	1858	1866	1880	1887	1894
		1901	1908	1915	1922	1930	1937	1944	1951	1958	1965	1972	1979	1986
		1993	2000	2007	2014	2056	2063	2070	2077	2084	2091	2098	2105	2112
		2119	2126	2133	2140	2147	2154	2161	2168	2175	2182	2189	2196	2203
		2210	2217	2224	2231	2238	2245	2252	2259	2266	2273	2280	2287	2294
		2301	2308	2315	2330	2337	2344	2351	2365	2372	2379	2386	2393	2400
		2407	2414	2421	2428	2435	2442	2449	2456	2463	2470	2477	2484	2491
		2512	2519	2526	2533	2540	2680	2687	2694	2715	2722	2729	2743	2750
		2757	2764	2771	2778	2785	2792	2800	2807	2815	2822	2829	2837	2845
		2853	2862	2870	2878	2885	2892	2899	2906	2913	2920	2927	2934	2941
		2948	2955	2962	2969	2976	2983	2990	2997	3004	3011	3018	3025	3032
		3039	3046	3053	3060	3067	3074	3109	3116	3130	3137	3144	3151	3158
		3179	12555#											
EHT110	071546	2021	12555#											
EHT111	071552	2028	2035	12555#										
EHT114	071556	2049	12555#											
EHT2	071536	2498	2505	2701	2708	12555#								
EHT223	071562	2547	12555#											
EHT256	071574	2736	12555#											
EHT336	071606	2323	3081	3095	3102	12555#								
EHT337	071620	3088	12555#											
EHT344	071632	3123	12555#											
EHT353	071644	3172	12555#											
EH1	100255	12555#												
EH110	100274	12555#												
EH111	100303	12555#												
EH114	100322	12555#												
EH223	100351	12555#												
EH256	100377	12555#												
EH336	100454	12555#												
EH337	100513	12555#												
EH344	100652	12555#												
EH353	101012	12555#												
EMS1	072026	12555#												
EMS10	072362	12555#												
EMS100	074602	12555#												
EMS101	074630	12555#												

```
EMS102  074656          12555#
EMS103  074705          12555#
EMS104  074714          12555#
EMS105  074743          12555#
EMS106  074772          12555#
EMS11   072424          12555#
EMS110  075011          12555#
EMS111  075040          12555#
EMS112  075051          12555#
EMS113  075135          12555#
EMS114  075165          12555#
EMS115  075211          12555#
EMS116  075216          12555#
EMS117  075246          12555#
EMS12   072435          12555#
EMS120  075270          12555#
EMS121  075305          12555#
EMS122  075350          12555#
EMS123  075405          12555#
EMS124  075450          12555#
EMS125  075502          12555#
EMS126  075545          12555#
EMS127  075610          12555#
EMS13   072476          12555#
EMS130  075647          12555#
EMS131  075705          12555#
EMS132  075745          12555#
EMS133  075771          12555#
EMS134  076014          12555#
EMS135  076056          12555#
EMS136  076120          12555#
EMS137  076145          12555#
EMS14   072507          12555#
EMS140  076202          12555#
EMS141  076223          12555#
EMS142  076251          12555#
EMS143  076266          12555#
EMS144  076314          12555#
EMS145  076324          12555#
EMS146  076353          12555#
EMS147  076372          12555#
EMS15   072520          12555#
EMS150  076407          12555#
EMS151  076426          12555#
EMS152  076452          12555#
EMS153  076476          12555#
EMS154  076520          12555#
EMS155  076534          12555#
EMS156  076554          12555#
EMS157  076572          12555#
EMS16   072542          12555#
EMS160  076607          12555#
EMS161  076624          12555#
        076654          12555#
        076722          12555#
EMS164  076750          12555#
```

```
EMS165  077014          12555#
EMS166  077030          12555#
EMS167  077055          12555#
EMS17   072572          12555#
EMS170  077102          12555#
EMS171  077131          12555#
EMS172  077164          12555#
EMS173  077204          12555#
EMS174  077221          12555#
EMS175  077227          12555#
EMS176  077255          12555#
EMS177  077304          12555#
EMS2    072075          12555#
EMS20   072632          12555#
EMS200  077326          12555#
EMS201  077343          12555#
EMS202  077411          12555#
EMS203  077441          12555#
EMS204  077454          12555#
EMS205  077502          12555#
EMS206  077514          12555#
EMS207  077541          12555#
EMS21   072655          12555#
EMS210  077566          12555#
EMS211  077577          12555#
EMS212  077615          12555#
EMS213  077640          12555#
EMS214  077663          12555#
EMS215  077701          12555#
EMS216  077722          12555#
EMS217  077752          12555#
EMS22   072700          12555#
EMS220  100005          12555#
EMS221  100025          12555#
EMS222  100075          12555#
EMS223  100126          12555#
EMS224  100203          12555#
EMS23   072714          12555#
EMS24   072742          12555#
EMS25   072771          12555#
EMS26   073006          12555#
EMS27   073027          12555#
EMS3    072112          12555#
EMS30   073040          12555#
EMS31   073050          12555#
EMS32   073077          12555#
EMS33   073126          12555#
EMS34   073154          12555#
EMS35   073225          12555#
EMS36   073254          12555#
EMS37   073303          12555#
EMS4    072155          12555#
EMS40   073331          12555#
EMS41   073360          12555#
EMS42   073406          12555#
EMS43   073435          12555#
```

```
EMS44   073464          12555#
EMS45   073537          12555#
EMS46   073602          12555#
EMS47   073631          12555#
EMS5    072220          12555#
EMS50   073660          12555#
EMS51   073707          12555#
EMS52   073735          12555#
EMS53   073747          12555#
EMS54   073761          12555#
EMS55   074003          12555#
EMS56   074032          12555#
EMS57   074107          12555#
EMS6    072250          12555#
EMS60   074135          12555#
EMS61   074150          12555#
EMS62   074177          12555#
EMS63   074215          12555#
EMS64   074243          12555#
EMS65   074261          12555#
EMS66   074327          12555#
EMS67   074377          12555#
EMS7    072315          12555#
EMS70   074424          12555#
EMS71   074436          12555#
EMS72   074451          12555#
EMS73   074461          12555#
EMS74   074476          12555#
EMS75   074515          12555#
EMS76   074523          12555#
EMS77   074554          12555#
EMTVEC= 000030            938#   3239*   3240*
EMT1    065432           1518   12555#
EMT10   065502           1567   12555#
EMT100  066500           1964   12555#
EMT101  066516           1971   12555#
EMT102  066534           1978   12555#
EMT103  066544           1985   12555#
EMT104  066556           1992   12555#
EMT105  066574           1999   12555#
EMT106  066604           2006   12555#
EMT107  066614           2013   12555#
EMT11   065506           1574   12555#
EMT110  066634           2020   12555#
EMT111  066646           2027   12555#
EMT112  066654           2034   12555#
EMT113  066662           2041   12555#
EMT114  066676           2048   12555#
EMT115  066704           2055   12555#
EMT116  066714           2062   12555#
EMT117  066724           2069   12555#
EMT12   065512           1581   12555#
EMT120  066734           2076   12555#
EMT121  066744           2083   12555#
EMT122  066754           2090   12555#
EMT123  066764           2097   12555#
```

EMT124	066774	2104	12555#
EMT125	067004	2111	12555#
EMT126	067014	2118	12555#
EMT127	067026	2125	12555#
EMT13	065520	1588	12555#
EMT130	067040	2132	12555#
EMT131	067052	2139	12555#
EMT132	067064	2146	12555#
EMT133	067076	2153	12555#
EMT134	067110	2160	12555#
EMT135	067122	2167	12555#
EMT136	067134	2174	12555#
EMT137	067146	2181	12555#
EMT14	065532	1595	12555#
EMT140	067156	2188	12555#
EMT141	067166	2195	12555#
EMT142	067176	2202	12555#
EMT143	067206	2209	12555#
EMT144	067216	2216	12555#
EMT145	067226	2223	12555#
EMT146	067236	2230	12555#
EMT147	067246	2237	12555#
EMT15	065540	1602	12555#
EMT150	067256	2244	12555#
EMT151	067266	2251	12555#
EMT152	067300	2258	12555#
EMT153	067312	2265	12555#
EMT154	067330	2272	12555#
EMT155	067346	2279	12555#
EMT156	067360	2286	12555#
EMT157	067372	2293	12555#
EMT16	065546	1609	12555#
EMT160	067404	2300	12555#
EMT161	067414	2307	12555#
EMT162	067426	2314	12555#
EMT163	067440	12555#	
EMT164	067450	2329	12555#
EMT165	067456	2336	12555#
EMT166	067464	2343	12555#
EMT167	067472	2350	12555#
EMT17	065556	1616	12555#
EMT170	067500	12555#	
EMT171	067510	2364	12555#
EMT172	067520	2371	12555#
EMT173	067530	2378	12555#
EMT174	067540	2385	12555#
EMT175	067552	2392	12555#
EMT176	067560	2399	12555#
EMT177	067566	2406	12555#
EMT2	065436	1525	12555#
EMT20	065566	1623	12555#
EMT200	067600	2413	12555#
EMT201	067612	2420	12555#
EMT202	067624	2427	12555#
EMT203	067636	2434	12555#
EMT204	067650	2441	12555#

EMT205	067666	2448	12555#
EMT206	067676	2455	12555#
EMT207	067706	2462	12555#
EMT21	065576	1630	12555#
EMT210	067720	2469	12555#
EMT211	067726	2476	12555#
EMT212	067742	2483	12555#
EMT213	067756	2490	12555#
EMT214	067766	2497	12555#
EMT215	070006	2504	12555#
EMT216	070020	2511	12555#
EMT217	070030	2518	12555#
EMT22	065606	1637	12555#
EMT220	070040	2525	12555#
EMT221	070050	2532	12555#
EMT222	070060	2539	12555#
EMT223	070070	2546	12555#
EMT224	070100	12555#	
EMT225	070102	12555#	
EMT226	070104	12555#	
EMT227	070106	12555#	
EMT23	065616	1644	12555#
EMT230	070110	12555#	
EMT231	070112	12555#	
EMT232	070114	12555#	
EMT233	070116	12555#	
EMT234	070120	12555#	
EMT235	070122	12555#	
EMT236	070124	12555#	
EMT237	070126	12555#	
EMT24	065626	1651	12555#
EMT240	070130	12555#	
EMT241	070132	12555#	
EMT242	070134	12555#	
EMT243	070136	12555#	
EMT244	070140	12555#	
EMT245	070142	12555#	
EMT246	070144	2679	12555#
EMT247	070154	2686	12555#
EMT25	065636	1658	12555#
EMT250	070166	2693	12555#
EMT251	070202	2700	12555#
EMT252	070210	2707	12555#
EMT253	070216	2714	12555#
EMT254	070234	2721	12555#
EMT255	070244	2728	12555#
EMT256	070254	2735	12555#
EMT257	070264	2742	12555#
EMT26	065646	1665	12555#
EMT260	070276	2749	12555#
EMT261	070310	2756	12555#
EMT262	070330	2763	12555#
EMT263	070340	2770	12555#
EMT264	070350	2777	12555#
EMT265	070370	2784	12555#
EMT266	070410	2791	12555#

EMT267	070422	2799	12555#
EMT27	065656	1672	12555#
EMT270	070440	2806	12555#
EMT271	070454	2814	12555#
EMT272	070472	2821	12555#
EMT273	070510	2828	12555#
EMT274	070526	2836	12555#
EMT275	070544	2844	12555#
EMT276	070556	2852	12555#
EMT277	070572	2861	12555#
EMT3	065444	1532	12555#
EMT30	065666	1679	12555#
EMT300	070610	2869	12555#
EMT301	070630	2877	12555#
EMT302	070652	2884	12555#
EMT303	070664	2891	12555#
EMT304	070676	2898	12555#
EMT305	070710	2905	12555#
EMT306	070722	2912	12555#
EMT307	070732	2919	12555#
EMT31	065676	1686	12555#
EMT310	070752	2926	12555#
EMT311	070764	2933	12555#
EMT312	070776	2940	12555#
EMT313	071010	2947	12555#
EMT314	071030	2954	12555#
EMT315	071042	2961	12555#
EMT316	071054	2968	12555#
EMT317	071066	2975	12555#
EMT32	065706	1693	12555#
EMT320	071076	2982	12555#
EMT321	071106	2989	12555#
EMT322	071116	2996	12555#
EMT323	071124	3003	12555#
EMT324	071134	3010	12555#
EMT325	071146	3017	12555#
EMT326	071160	3024	12555#
EMT327	071176	3031	12555#
EMT33	065716	1700	12555#
EMT330	071210	3038	12555#
EMT331	071220	3045	12555#
EMT332	071232	3052	12555#
EMT333	071244	3059	12555#
EMT334	071262	3066	12555#
EMT335	071300	3073	12555#
EMT336	071320	2322	3080 12555#
EMT337	071330	3087	12555#
EMT34	065726	1707	12555#
EMT340	071340	3094	12555#
EMT341	071352	3101	12555#
EMT342	071360	3108	12555#
EMT343	071372	3115	12555#
EMT344	071404	3122	12555#
EMT345	071416	3129	12555#
EMT346	071426	3136	12555#
EMT347	071442	3143	12555#

```
EMT35   065736            1714   12555#
EMT350  071454            3150   12555#
EMT351  071472            3157   12555#
EMT352  071502            3164   12555#
EMT353  071506            3171   12555#
EMT354  071516            3178   12555#
EMT36   065746            1721   12555#
EMT37   065756            1728   12555#
EMT4    065452            1539   12555#
EMT40   065766            1735   12555#
EMT41   065774            1742   12555#
EMT42   066004            1749   12555#
EMT43   066016            1756   12555#
EMT44   066026            1763   12555#
EMT45   066036            1770   12555#
EMT46   066046            1777   12555#
EMT47   066056            1784   12555#
EMT5    065460            1546   12555#
EMT50   066064            1791   12555#
EMT51   066074            1798   12555#
EMT52   066106            1805   12555#
EMT53   066124            1813   12555#
EMT54   066142            1820   12555#
EMT55   066152            1827   12555#
EMT56   066164            1834   12555#
EMT57   066202            1841   12555#
EMT6    065466            1553   12555#
EMT60   066212            1849   12555#
EMT61   066230            1857   12555#
EMT62   066250            1865   12555#
EMT63   066264           12555#
EMT64   066266            1879   12555#
EMT65   066306            1886   12555#
EMT66   066326            1893   12555#
EMT67   066340            1900   12555#
EMT7    065474            1560   12555#
EMT70   066352            1907   12555#
EMT71   066362            1914   12555#
EMT72   066372            1921   12555#
EMT73   066410            1929   12555#
EMT74   066426            1936   12555#
EMT75   066436            1943   12555#
EMT76   066456            1950   12555#
EMT77   066466            1957   12555#
ENRGDT  065432           12555#
EQUALS  063573           12555#
ERR     = 040000          1013#   9269    9271    9287    9321
ERRNMB  033734            7828#   7829#   7832    7844    7849    7941#
ERRTYP  033176            7807#  12084
ERRVEC= 000004             931#   3253    3254#   3265#   3533    3534    3535#   3536#   3547#   3548#   3552#   3553#   3579
                          3580    3581#   3582#   3586#   3587#   3598#   3599#   8841    8842    8846#   8847#   8878#   8879#
                          8914    8915    8919#   8920#   8945#   8946#   8957    8958    8959#   8960#   8966#   8974#   8975#
                          8989    8990    8991#   8992#   9016#   9017#   9864    9865    9868#   9869#   9902#   9903#  10150
                         10151   10152#  10153#  10164#  10165#  11973   11974#  11976#  11979#
ERTY00  033742            7814    7946#
ERTY01  033750            7822    7947#
```

```
GETBUF  001330          1420#  8844
GETINX  001506          1491#  3623*  3624*  4479*  4480*  4994*  4995*  8047*  8048*  8097*  8098*  8790  8845
GETSTS  037432          3693   3861   4029   4199   4359   4547   4715   4875   5063   5222   5381   5542  5707
                        5852   5997   6141   6222   6309   6383   6456   6510   6603   6657   6761   6937  6991
                        7077   7215   7363   7599   8024   8075   8118   8216   8786#
GNS     = ****** U      1262   3231   7766   7773   12439  12440  12441  12442  12443  12444  12446  12448 12449
                       12450  12451  12452  12453
GO      = 000001         954#  3678   3715   3762   3846   3883   3930   4014   4052   4100   4184   4221  4268
                        4344   4381   4431   4532   4569   4616   4700   4737   4784   4860   4897   4947  5048
                        5085   5132   5207   5244   5291   5366   5403   5450   5527   5565   5612   5690  5743
                        5835   5888   5980   6033   6118   6202   6292   6353   6439   6500   6586   6647  6716
                        6732   6747   6795   6811   6918   6981   7060   7114   7198   7252   7346   7399  7519
                        7582   7635   7714   8059   8108   8220   8253   8287   8321   9002   9113   9125  9130
                        9146   9236   9239   11193
GTSWR   = 104407        3286   3507   12446#
HCE     = 000200        1034#  1043   7466   7476   7663   7673   8366   9564   9569   9584   9586   9599  9614
                       11199  11229  11232  11242  11245  12555
HCI     = 002000        1114#  9562   11195
HCRC    = 000400        1033#  1043   8366   9571   11199  11203  11206  11242  11245
HELP    = 103566        3368   12555#
HELPQS  = 063603        3362   12555#
HT      = 000011         841#  7870   11910  11951
IAE     = 002000        1031#  9468   9765   9780   9806   9952   9969   10468  10522  10525  11026 11041 12555
IDXMSK= 000077          1190#  8858   8861   8931   8934
IE      = 000100        1204#  1237#
ILF     = 000001        1041#  9340   9342   10362  10422  10426  10468  10489  10492  10982  10999 11002 12555
ILF02 = 000002           960#
ILF24 = 000024           970#
ILF26 = 000026           971#
ILF30 = 000030           973#
ILF32 = 000032           974#
ILF34 = 000034           975#
ILF36 = 000036           976#
ILF40 = 000040           977#
ILF42 = 000042           978#
ILF44 = 000044           979#
ILF46 = 000046           980#
ILF54 = 000054           983#
ILF56 = 000056           984#
ILF64 = 000064           987#
ILF66 = 000066           988#
ILF74 = 000074           991#
ILF76 = 000076           992#
ILR     = 000002        1040#  10362  10408  10412  10676  10680  10683  10982  10985  10988
ILRG50= 000050          1176#
ILRG52= 000052          1177#
ILRG54= 000054          1178#
ILRG56= 000056          1179#
ILRG60= 000060          1180#
ILRG62= 000062          1181#
ILRG64= 000064          1182#
ILRG66= 000066          1183#
ILRG70= 000070          1184#
ILRG72= 000072          1185#
ILRG74= 000074          1186#
ILRG76= 000076          1187#
```

```
IOTVEC= 000020            936#   3237*   3238*
IPCK0 = 000001           1241#
IPCK1 = 000002           1240#
IPCK2 = 000004           1239#
IPCK3 = 000010           1238#
IR    = 000100           1218#  10225
IVC   = 010000           1147#   6956    6960    7010    7014    9383    9384    9987   10110   10536   10543   10546   10615
                        10965   10969   11397   11478   12555
LBC   = 002000           1149#  11083   11101   11106
LBT   = 002000           1017#   9543    9767    9769
LF    = 000012            842#   7868   11652   11945   11951   12555
LS    = 000004           1096#  10249   10801
LSC   = 004000           1148#  11083   11117   11120
LST   = 000002           1097#  10249   10801
MCLK  = 004000           1069#
MCPE  = 020000           1198#   9113    9130    9143    9146   10880   10883
MDF   = 000100           1074#
MDPE  = 000400           1215#   9639   11289   11292
MEDENB  001474           1479#   3519*   8058*   8181    8437*
MFGFIL  101546           8203    8406    8419*   8426*   8487   12555*
MI    = 000004           1077#
MIXED   065004          12555#
MOC   = 000400           1072#
MOH   = 020000           1104#
MOL   = 010000           1015#   8358   10048   10058   10059   10064   10069   10097   10348   10351   10512   10589   10590
                        10597   10602   10761   10942   11103   11356   11357   11379   11384   11455   11456   11460   11465
MRD   = 002000           1070#
MS    = 000040           1075#
MSC   = 000002           1078#
MSE   = 100000           1155#   7425    8406    8629
MSER  = 000200           1073#
MUR   = 001000           1071#
MWD   = 000010           1095#  10250   10802
MWP   = 000010           1076#
MXF   = 001000           1214#   9528   10903   10917   10920
NOTMSK= 115760           1043#
NED   = 010000           1211#   3318    3588    9095    9099    9877
NEM   = 004000           1212#   9513
NOP   = 000000            959#
NOTEX   064647           3326   12555#
NSA   = 100000           1102#
OCC   = 100000           1083#
OFD   = 000200           1115#   6866    6868
OFFSET= 000014            965#   6732    6795
OM    = 000001           1022#  10589   10645   10648   10760   11304   11307
ONES    065044           4517    4685    4845   12555#
OPE   = 020000           1146#
OPI   = 020000           1028#   9367   10042   10066   10362   10380   10384   10468   10505   10508   10599   10937   10940
                        11381   11462   12555
OR    = 000200           1217#
PACACK= 000022            969#   8253
PAKACK= 000022            968#    969    8059
PAR   = 000010           1038#   8369    9158    9162    9174    9934   10468   10473   10478   10894   10899
PAT   = 000020           1220#
PDA   = 000400           1090#
PGE   = 002000           1213#
```

```
PGM     = 001000          1018#  10760
PHA     = 000200          1091#
PIP     = 020000          1014#  8106   8284   10058  10079  10084  10589  10660  10665  10760  11356  11362  11367
                          11455  11492  11497
PIRQ    = 177772           848#
PIRQVE= 000240             942#
PLFS    = 002000          1088#
PRIERR  040512            3738   3781   3906   3949   4075   4119   4244   4286   4404   4449   4592   4635   4760
                          4802   4920   4965   5108   5150   5267   5309   5426   5468   5588   5630   5719   5762
                          5864   5907   6009   6052   6152   6234   6321   6375   6468   6522   6615   6669   6773
                          6835   6949   7003   7089   7133   7227   7271   7375   7418   7611   7654   9061#
PROMPT  063575            3443   3452   12555#
PRO     = 000000           865#
PR1     = 000040           866#
PR2     = 000100           867#
PR3     = 000140           868#  11628
PR4     = 000200           869#
PR5     = 000240           870#
PR6     = 000300           871#  3224   3514   3536   3582   8847   8920   8960   8992   9869   10153  11634
PR7     = 000340           872#
PS      = 177776           845#  846
PSEL    = 002000          1200#
PSW     = 177776           846#
PUT       037766          3685   3721   3764   3853   3889   3932   4021   4058   4102   4191   4227   4269   4351
                          4387   4432   4539   4575   4618   4707   4743   4785   4867   4903   4948   5055   5091
                          5133   5214   5250   5292   5373   5409   5451   5534   5571   5613   5699   5745   5844
                          5890   5989   6035   6133   6214   6301   6355   6448   6502   6595   6649   6739   6753
                          6802   6818   6910   6929   6983   7069   7116   7207   7254   7355   7401   7528   7591
                          7637   7723   8062   8111   8221   8254   8288   8322   8908#
PUTBUF  001400            1447#  8192   8441*  8917
PUTINX  001535            1496#  3679   3716   3847   3884   4015   4053   4185   4222   4345   4382   4533   4570
                          4701   4738   4861   4898   5049   5086   5208   5245   5367   5404   5528   5566   5691
                          5836   5981   6120   6203   6293   6440   6587   6733   6748   6796   6813   6908*  6909*
                          6920   7061   7199   7347   7520   7583   7715   8060*  8061*  8109*  8110*  8205   8918
                          9165
PWRVEC= 000024             937#  3243*  3244*  12458*  12459*  12468*  12474*  12486*  12487*
QSTMRK  063601            3357   3462   12555#
RCLSTS  050452            8132   8307   10459#
RD      = 000070           989#
RDCHR   = 104411          3349   3363   3373   3376   3444   3453   12311  12449#
RDLIN   = 104412         12381   12450#
RDOCT   = 104413          3387   3403   3424   12451#
RDY     = 000200          1203#  9002   9003   9109   9112   10200
READY     007124          3354   3511#  7741   11645
RECAL   = 000006           962#  8108   8287
RESREG= 104415            7937   12453#
RESVEC= 000010             932#
REX     = 010000          1086#
RG      = 040000          1084#
RGOTPT  065004           12555#
RH      = 000072           990#  3762   3930   4100   4616   5743   5888   6033   6118   6202   6353   6500   6647
                          6811   6981   7114   7252   7399   7635   8321
RIP     = 000020           967#
RLEASE= 000012             964#
RMAS    = 000016          1164#
RMASI     001346          1430#  10781  12555
```

```
RMASO   001416          1457#
RMBA  = 000004          1246#  3541#  3542   3718   3886   4055   4224   4384   4572   4740   4900   5088   5247
                        5406   5568   5696   5841   5986   6124   6208   6298   6445   6592   6750   6815   6926
                        7066   7204   7352   7525   7588   7719   8210
RMBAE = 000050          1249#
RMBAI   001334          1425#  4475   4990   9715   9717   10211  12555
RMBAO   001404          1452#  3661#  3763#  3829#  3931#  3997#  4101#  4167#  4327#  4515#  4617#  4683#  4843#
                        5031#  5190#  5349#  5508#  5673#  5744#  5818#  5889#  5963#  6034#  6117#  6201#  6275#
                        6354#  6422#  6501#  6569#  6648#  6715#  6812#  6869#  6890#  6982#  7043#  7115#  7181#
                        7253#  7320#  7400#  7509#  7556#  7636#  7705#  8203#  8621   8708   9711   9714
RMCC  = 000036          1171#
RMCCI   001366          1438#
RMCCO   001436          1465#
RMCS1 = 000000          1160#  1244#  3320   3538#  3584   3683   3719   3851   3887   4019   4056   4189   4225
                        4349   4385   4537   4573   4705   4741   4865   4901   5053   5089   5212   5248   5371
                        5407   5532   5569   5697   5842   5987   6126   6209   6299   6446   6593   6737   6751
                        6800   6816   6927   7067   7205   7353   7526   7589   7721   8060   8109   8211   8849
                        8922   9001   9875
RMCS1I  001330          1423#  9088   9090   9092   9109   9111   9114   9125   9129   9131   9143   9145   9147
                        9236   9238   9249   9292   9698   10198  10747  10880  10882  10884  10903  12555
RMCS1O  001400          1450#  3662#  3678#  3715#  3762#  3830#  3846#  3883#  3930#  3998#  4014#  4052#  4100#
                        4168#  4184#  4221#  4268#  4328#  4344#  4381#  4431#  4516#  4532#  4569#  4616#  4684#
                        4700#  4737#  4784#  4844#  4860#  4897#  4947#  5032#  5048#  5085#  5132#  5191#  5207#
                        5244#  5291#  5350#  5366#  5403#  5450#  5509#  5527#  5565#  5612#  5674#  5690#  5743#
                        5819#  5835#  5878#  5964#  5980#  6033#  6118#  6202#  6276#  6292#  6353#  6423#  6439#
                        6500#  6570#  6586#  6647#  6716#  6732#  6747#  6795#  6811#  6891#  6918#  6981#  7044#
                        7060#  7114#  7182#  7198#  7252#  7321#  7346#  7399#  7510#  7519#  7557#  7582#  7635#
                        7706#  7714#  8059#  8108#  8220#  8253#  8287#  8321#  8467   8625   9213   9729   11171
                        11191
RMCS2 = 000010          1247#  3315#  3316#  3317   3543#  3544   3585   8848   8921   9874#  9876   10155#  10157#
RMCS2I  001340          1427#  4458   4466   4468   4974   4982   4984   9070   9095   9097   9098   9284   9355
                        9456   9499   9514   9529   9640   9712   10220  10917  10919  10921  11289  11291  11293
                        11343  11345  11347  12555
RMCS2O  001410          1454#
RMCS3 = 000052          1250#
RMDA  = 000006          1161#  3680   3848   4016   4186   4346   4534   4702   4862   5050   5209   5368   5529
                        5692   5837   5982   6121   6204   6294   6441   6588   6734   6797   6922   7062   7200
                        7348   7521   7584   7716   8206
RMDAI   001336          1426#  9811   12555
RMDAO   001406          1453#  3658#  3826#  3994#  4164#  4324#  4512#  4680#  4840#  5028#  5187#  5346#  5505#
                        5670#  5675   5795#  5796   5815#  5820   5940#  5941   5960#  6114#  6198#  6272#  6277
                        6399#  6400   6419#  6424   6546#  6547   6566#  6712#  6717   6887#  6892   7040#  7178#
                        7317#  7553#  8199#  8380#  8381   8383   8401   8409#  8461   8588#  8624   9737   9738
                        9957   9959   9963   11031  11033  11035
RMDB  = 000022          1248#  3545#  3546   4479   4994   8797
RMDBI   001352          1432#  4486   5001
RMDBO   001422          1459#
RMDC  = 000034          1170#  3681   3849   4017   4187   4347   4535   4703   4863   5051   5210   5369   5530
                        5693   5838   5983   6122   6205   6295   6442   6589   6735   6798   6923   7063   7201
                        7349   7522   7585   7717   8207
RMDCI   001364          1437#  9823   12555
RMDCO   001434          1464#  3657#  3825#  3993#  4163#  4323#  4511#  4679#  4839#  5027#  5186#  5345#  5504#
                        5525   5526#  5564#  5669#  5814#  5959#  5965   6085#  6086   6113#  6197#  6271#  6418#
                        6565#  6571   6692#  6693   6711#  6886#  7039#  7177#  7316#  7552#  8200#  8460   8587#
                        8623   9739   9954   11028
RMDS  = 000012          1162#  8047   8097
```

```
RMDSI   001342          1428#  8056   8106   8251   8284   8358   9127   9223   9225   9270   9287   9321   9325
                        9381   9408   9543   9768   9993  10048  10057  10064  10068  10070  10079  10083  10085
                       10094  10096  10098  10108  10112  10114  10293  10334  10336  10338  10348  10350  10352
                       10512  10550  10588  10597  10601  10603  10613  10617  10618  10629  10631  10633  10645
                       10647  10649  10660  10664  10666  10759  10942  10944  10973  11103  11169  11304  11306
                       11308  11355  11362  11366  11368  11379  11383  11385  11395  11399  11401  11454  11460
                       11464  11466  11476  11480  11481* 11482  11492  11496  11498  12555
RMDSO   001412          1455#
RMDT  = 000026          1167#  3623
RMDTI   001356          1434#  3625*  3630   3632   3634   3636  12555
RMDTO   001426          1461#
RMEC1 = 000044          1174#
RMEC1I  001374          1441#  8709  12555
RMEC10  001444          1468#
RMEC2 = 000046          1175#  8792
RMEC2I  001376          1442#  8730   8791  10261  10825  12555
RMEC20  001446          1469#
RMER1 = 000014          1163#  9166
RMER1I  001344          1429#  7142   7150   7151   7280   7288   7289   7431   7439   7440   7466   7474   7475
                        7663   7671   7672   8365   8698   8700   9158   9173   9175   9265   9341   9368   9411
                        9426   9469   9546   9572   9587   9602   9655   9674   9677   9765   9780   9795   9806
                        9934   9941   9968  10042  10045  10066  10238  10362  10366  10368  10369  10380  10382
                       10383  10394  10396  10397  10408  10410  10411  10422  10424  10425  10468  10473  10477
                       10479  10489  10491  10493  10505  10507  10509  10522  10524  10526  10599  10676  10680
                       10682  10684  10694  10696  10698  10708  10712  10714  10772  10894  10898  10900  10937
                       10939  10941  10982  10985  10987  10989  10999  11001  11003  11013  11015  11017  11040
                       11130  11133  11137  11139  11148  11150  11152  11162  11164  11165  11199  11203  11205
                       11207  11216  11218  11220  11229  11231  11233  11242  11244  11246  11264  11266  11268
                       11274  11330  11332  11334  11381  11462  11573  12555
RMER10  001414          1456#  9162
RMER2 = 000042          1173#
RMER2I  001372          1440#  6956   6958   6959   7010   7012   7013   7446   7455   7456   8367   8372   9160
                        9267   9385   9441   9617   9936   9987   9990  10001  10029  10032  10081  10110  10283
                       10475  10536  10543  10545  10547  10560  10562  10564  10574  10576  10578  10615  10662
                       10710  10834  10896  10965  10968  10970  11054  11058  11083  11087  11089  11091  11101
                       11105  11107  11117  11119  11121  11135  11317  11319  11321  11364  11397  11478  11494
                       11508  11514  11516  11518  11528  11530  11532  11576  12555
RMER20  001442          1467#
RMLA  = 000020          1165#
RMLAI   001350          1431#  12555
RMLA0   001420          1458#
RMMR1 = 000024          1166#   6908   6921
RMMR1I  001354          1433#  10248  10800  12555
RMMR10  001424          1460#   6907*  6919*
RMMR2 = 000040          1172#
RMMR2I  001370          1439#  10270  10812  12555
RMMR20  001440          1466#
RMOF  = 000032          1169#   3682   3850   4018   4188   4348   4536   4704   4864   5052   5211   5370   5531
                        5694    5839   5984   6123   6206   6296   6443   6590   6736   6799   6924   7064   7202
                        7350    7523   7586   7718   8209
RMOFI   001362          1436#   8702   8704   9562   9672  11195  11271  12555
RMOF0   001432          1463#   3659*  3827*  3995*  4165*  4325*  4513*  4681*  4841*  5029*  5188*  5347*  5506*
                        5671*   5816*  5961*  6115*  6199*  6273*  6420*  6567*  6713*  6866   6868*  6888*  7041*
                        7113*   7179*  7251*  7318*  7507*  7554*  7703*  8202*  8631   9961  11037  12555
RMR   = 000004          1039#  10362  10394  10398  10676  10694  10697  10982  11013  11016
RMSN  = 000030          1168#
```

```
RMSNI   001360          1435#  12555
RMSNO   001430          1462#
RMWC    = 000002        1245#   3539*   3540    3717    3885    4054    4223    4383    4571    4739    4899    5087    5246
                        5402    5567    5695    5840    5985    6125    6207    6297    6444    6591    6749    6814    6925
                        7045    7203    7351    7524    7587    7720    8208
RMWCI   001332          1474#   8747    9696    9708    9726    9790   12555
RMWCO   001402          1451#   3660*   3828*   3996*   4051*   4099*   4166*   4326*   4514*   4682*   4842*   5030*   5189*
                        5348*   5507*   5672*   5817*   5962*   6116*   6200*   6274*   6421*   6568*   6714*   6889*   7042*
                        7180*   7319*   7508*   7555*   7704*   8201*   8463    8622    8748    9709    9727
RQA     = 100000        1124#  10271   10813
RQB     = 040000        1125#  10271   10813
RTC     = 000016         966#
SADMSK  = 000037        1008#
SAVREG  = 104414        7808   12452#
SA1     = 000001        1003#
SA16    = 000020         999#
SA2     = 000002        1002#
SA4     = 000004        1001#
SA8     = 000010        1000#
SC      = 100000        1196#   9113    9130    9146   10199
SCTMSG  063462         12555#
SCTMSK  = 003700        1059#
SCO     = 000100        1057#
SC1     = 000200        1056#
SC2     = 000400        1055#
SC3     = 001000        1054#
SC4     = 002000        1053#
SEARCH  = 000030         972#   9289    9690
SECERR  041344          3752    3795    3920    3963    4089    4133    4258    4300    4418    4606    4649    4774    4816
                        4934    5122    5164    5281    5323    5440    5482    5602    5644    5733    5776    5878    5921
                        6023    6066    6166    6248    6335    6389    6482    6536    6629    6683    6787    6849    6963
                        7017    7103    7155    7241    7293    7389    7486    7625    7683    9209#
SEEK    = 000004         961#   3678    3846    4014    4184    4344    4532    4700    4860    5048    5207    5366    5527
SEKSTS  045420          3706    3874    4042    4212    4372    4560    4728    4888    5076    5235    5394    5555    9923#
SHUT    057364          7789   11642#
SHUT2   057444         11657#  12152   12235
SIZCLK  040204          3494    8956#
SKI     = 040000        1145#   9953    9956    9972   10002   10004   10008   10019   10081   10284   10536   10560   10563
                       10662   11027   11030   11044   11055   11057   11058   11060   11072   11364   11494   11509   11528
                       11531
SNGPRT  = 020024        1107#   3630    3634    3638
STACK   = 001100         836#   3235    3528    3568    3612    3650    3818    3986    4156    4316    4504    4672    4832
                        5020    5179    5338    5497    5662    5807    5952    6097    6182    6264    6411    6558    6704
                        6879    7032    7170    7308    7544
STAMDA  006144          3304    3339#
START   005324          1282    3219#   3557   11651
STCDRV  056416         11447#
STKLMT  = 177774         847#
STOP    057334         11625#  11967
STSD1   101436         12555#
STSD2   101454         12555#
STSD3   101474         12555#
STSD4   101506         12555#
STSF    101537         12555#
STSM1   101071         12555#
STSM2   101150         12555#
```

```
STSH3   101241          12555#
STSH4   101301          12555#
SWR     001154           1335#  3233   3255*  3257   3263*  3270*  3284   3505   7809   11968  11982  11985  12000
                        12007  12075  12082  12094  12098  12155  12194  12249* 12466  12479*
SWREG   000176           1264#  3263   3284   3505   12155  12194  12217
SW0   = 000001            900#
SW00  = 000001            890#   900
SW01  = 000002            889#   899
SW02  = 000004            888#   898
SW03  = 000010            887#   897
SW04  = 000020            886#   896
SW05  = 000040            885#   895
SW06  = 000100            884#   894
SW07  = 000200            883#   893
SW08  = 000400            882#   892
SW09  = 001000            881#   891
SW1   = 000002            899#
SW10  = 002000            880#
SW11  = 004000            879#
SW12  = 010000            878#
SW13  = 020000            877#   7809
SW14  = 040000            876#
SW15  = 100000            875#
SW2   = 000004            898#
SW3   = 000010            897#
SW4   = 000020            896#
SW5   = 000040            895#
SW6   = 000100            894#
SW7   = 000200            893#
SW8   = 000400            892#
SW9   = 001000            891#
TADMSK= 003400           1007#
TAG   = 020000           1126#
TAGADR= 001114           1307#  1314
TAP   = 040000           1103#
TA1   = 000400            998#
TA2   = 001000            997#
TA4   = 002000            996#
TBITVE= 000014            933#
TIMOUT  040326          3694   3729   3772   3862   3897   3940   4030   4066   4110   4200   4235   4277   4360
                        4395   4440   4548   4583   4626   4716   4751   4793   4876   4911   4956   5064   5099
                        5141   5223   5258   5300   5382   5417   5459   5543   5579   5621   5710   5753   5855
                        5898   6000   6043   6143   6225   6312   6366   6459   6513   6606   6660   6764   6826
                        6940   6994   7080   7124   7218   7262   7366   7409   7534   7602   7645   7729   8119
                        8297   8331   8985#
TKVEC = 000060            940#  12132* 12133*
TPVEC = 000064            941#
TRAPVE= 000034            939#  3241*  3242*
TRE   = 040000           1197#  9113   9130   9146   9291   9293   9698
TRTVEC= 000014            934#
TST   = 010000           1127#  10272  10814
TSTNMB  033732           7820#  7821*  7823   7940#
TSTPRP  034000           3669   3837   4005   4175   4335   4523   4691   4851   5039   5198   5357   5516   5681
                         5826   5971   6104   6188   6283   6344   6430   6491   6577   6638   6723   6898   6972
                         7051   7189   7328   7513   7564   7708   7988#
TSTQUE  001450           1475#  3477   3478*  3518   3530   3570   3614   3652   3820   3988   4158   4318   4506
```

		4674	4834	5022	5181	5340	5499	5664	5809	5954	6099	6184	6266	6413
		6560	6706	6881	7034	7172	7310	7546	7736	7738*	7742*	9871	10156	10223
		10784												
TST1	007162	3523#	12027											
TST10	012716	4313#	12034											
TST11	013636	4501#	12035											
TST12	014436	4669#	12036											
TST13	015212	4829#	12037											
TST14	016130	5017#	12038											
TST15	016704	5176#	12039											
TST16	017460	5335#	12040											
TST17	020234	5494#	12041											
TST2	007360	3565#	12028											
TST20	021036	5659#	12042											
TST21	021562	5804#	12043											
TST22	022306	5949#	12044											
TST23	023030	6094#	12045											
TST24	023326	6179#	12046											
TST25	023624	6261#	12047											
TST26	024352	6408#	12048											
TST27	025100	6555#	12049											
TST3	007544	3609#	12029											
TST30	025626	6701#	12050											
TST31	026520	6876#	12051											
TST32	027326	7029#	12052											
TST33	030050	7167#	12053											
TST34	030572	7305#	12054											
TST35	031750	7541#	12055											
TST4	007726	3647#	12030											
TST5	010526	3815#	12031											
TST6	011326	3983#	12032											
TST7	012142	4153#	12033											
TYPBN =	104406	7930	12444#											
TYPDS =	104405	7770	7777	7924	12443#									
TYPE =	104401	3278	3298	3326	3348	3351	3357	3362	3365	3368	3372	3375	3383	3386
		3392	3397	3402	3408	3418	3423	3429	3442	3443	3446	3452	3455	3462
		7764	7771	7778	7813	7814	7822	7831	7837	7846	7884	7890	7894	7907
		7913	7915	7933	7935	11646	11720	11786	11855	11915	12077	12085	12149	12161
		12215	12216	12219	12230	12240	12251	12270	12318	12324	12329	12333	12338	12339
		12341	12344	12348	12439#	12488								
TYPOC =	104402	3385	7927	12218	12440#									
TYPON =	104404	12442#												
TYPOS =	104403	3328	3399	3420	7817	7825	7834	7840	12441#					
UBUSQS	063772	3372	12555#											
UNS =	040000	1027#	10239	10362	10366	10370	10676	10708	10713	11130	11133	11138		
UNTMSK=	000007	1228#	9071	9073										
UPE =	020000	1210#	9455											
USE =	040000	1156#	7425	8406	8629									
USRFIL	102556	8420*	8427*	8578	8580	12555#								
U0 =	000001	1225#												
U1 =	000002	1224#												
U2 =	000004	1223#												
VV =	000100	1021#	8056	8251	9381	9993	10058	10059	10108	10113	10334	10337	10550	10589
		10590	10613	10619	10760	10944	10973	11356	11357	11395	11400	11455	11456	11476
		11481												
WC =	000040	1093#	10249	10801										

```
WCD     = 000050            981#
WCE     = 040000           1209#  4458   4467   4974   4983   9354  12555
WCEHI   = 010000           1234#
WCELO   = 004000           1235#
WCF     = 000040           1036#  1043   9425  11330  11333
WCH     = 000052            982#  4268   4431   4784   4947   5132   5291   5450   5612
WD      = 000060            985#
WH      = 000062            986#  3662   3715   3830   3883   3998   4052   4168   4221   4328   4381   4516   4569
                           4684   4737   4844   4897   5032   5085   5191   5244   5350   5403   5509   5565   5674
                           5690   5819   5835   5964   5980   6276   6292   6423   6439   6570   6586   6716   6747
                           6891   6918   7044   7060   7182   7198   7321   7346   7510   7519   7557   7582   7706
                           7714  11193
WLE     = 004000           1030#  1043   9406   9410   9423   9438   9453  11130  11162  11166  11174  12555
WRL     = 004000           1016#  9408  11169
XDPMG     064562           3298  12555#
XSIZ      006014           3308#  3449
ZEROS     065106           3663   3831   3999   4169   4329   5033   5192   5351   5510   7045   7183   7322   7558
                          12555#
$APTHD    001100           1294   1300#
$ASTAT=   ******  U       12531  12546
$ATYC     063240          12502  12504#
$ATY1     063214          12500#
$ATY3     063222          11900  12501#
$ATY4     063232          12090  12503#
$AUTOB    001150           1332#  3288*  3509*  12208  12365
$BASE     001276           1401#  3314   3384   3394*  3529   3569   3613   3651   3819   3987   4157   4317   4505
                           4673   4833   5021   5190   5339   5498   5663   5808   5953   6098   6183   6265   6412
                           6559   6705   6880   7026   7171   7309   7545   8843   8916   8993   9870  10154  12555
$BDADR    001136           1327#  4475*  4476*  4489   4990*  4991*  5004   8758*  12555
$BDDAT    001142           1329#  4468*  4486*  4493   4984*  5001*  5009   6958*  7012*  7150*  7288*  7439*  7455*
                           7474*  7671*  8756*  9070*  9071*  9074   9092*  9098*  9114*  9131*  9147*  9175*  9225*
                           9226*  9238*  9239*  9249*  9250*  9252   9270*  9271*  9272   9292*  9293*  9294   9325*
                           9326*  9327   9341*  9342*  9343   9355*  9356*  9368*  9369*  9385*  9386*  9411*  9412*
                           9426*  9427*  9441*  9442*  9456*  9457*  9469*  9470*  9499*  9500*  9514*  9515*  9529*
                           9530*  9546*  9547*  9572*  9573*  9587*  9588*  9602*  9603*  9617*  9618*  9640*  9641*
                           9655*  9656*  9677*  9678*  9696*  9717*  9768*  9769*  9770   9795*  9796*  9797   9811*
                           9812   9823*  9824   9941*  9968*  9969*  9970   9990*  10001*  10002*  10008  10019  10032*
                          10045*  10070*  10084*  10085*  10098*  10114*  10198*  10199*  10201  10211*  10220*  10228  10236*
                          10239*  10248*  10249*  10251  10261*  10270*  10271*  10273  10283*  10284*  10293*  10294*  10296
                          10338*  10352*  10368*  10382*  10396*  10410*  10424*  10479*  10493*  10509*  10526*  10547*  10564*
                          10578*  10603*  10618*  10633*  10649*  10666*  10684*  10698*  10714*  10747*  10748*  10750  10759*
                          10760*  10762  10772*  10781*  10788*  10791  10800*  10801*  10803  10812*  10813*  10815  10825*
                          10834*  10884*  10900*  10921*  10941*  10970*  10989*  11003*  11017*  11040*  11041*  11042  11054*
                          11055*  11091*  11107*  11121*  11139*  11152*  11164*  11207*  11220*  11233*  11246*  11268*  11293*
                          11308*  11321*  11334*  11347*  11368*  11385*  11401*  11466*  11482*  11498*  11518*  11532*  11588*
                          11603*  12555
$BELL     001212           1352#  12077  12110  12161  12358
$BIN      057622          11716*  11719*  11720  11727#
$CDW1     001302           1403#
$CDW2     001304           1404#
$CHARC    060554          11917*  11927*  11934  11943*  11948#
$CKSWR    061666          12194#  12448
$CMTAG    001114           1315#   3230    3231   3239   3245   3246   3247
$CM3    = 000000           1345#
$CM4    = 000005           1345#   1346#  1347#  1348#  1349#  1350#
$CNTLC    062564          12149  12230  12358#
```

```
$CNTLG  062576          12215  12360#
$CNTLU  062571          12240  12333  12359#
$CPUOP  001250           1375#
$CRLF   001217           1354#  7778   7813   7846   7884   7894   7907   7915   7935  11916  11951  12085  12110
                        12251  12338  12358
$DBLK   060040          11752  11786  11794#
$DDW0   001306           1405#
$DDW1   001310           1406#
$DDW2   001312           1407#
$DDW3   001314           1408#
$DDW4   001316           1409#
$DDW5   001320           1410#
$DDW6   001322           1411#
$DDW7   001324           1412#
$DEVCT  001232           1366#
$DEVM   001300           1402#  3310*  3311*  3323*  3441*  3450*  3468*  3469   3476
$DOAGN  033166           7760   7781   7787#
$DTBL   060030          11755  11790#
$ENDAD  033156           1273   3276   7783#  12105
$ENDCT  033022           3245   7762#
$ENULL  033172           7790#
$ENV    001242           1371#  3282   3303   3503  11895  12087  12509  12533
$ENVM   001243           1372#  3268   3308  11897  11902  12511
$EOP    032766           3558   7752#  11649
$EOPCT  033014           3245*  7759#  7763
$EOSP   032732           3601   3641   7734#
$ERFLG  001117           1318#  11956  11996  11998  12004*  12026  12072*  12110
$ERMAX  001131           1324#  3248*  11998  12021*  12026
$ERROR  061170           3239   12070#
$ERRPC  001132           1325#  7838   12079*  12080*  12081  12110
$ERRTB  001564           1514#  7853
$ERTTL  001126           1322#  7775   7779*  12078*  12110
$ESCAP  001210           1351#  3247*  12020*  12101  12103  12110
$ETABL  001242           1370#
$ETEND  001326           1306   1413#
$FATAL  001224           1363#  12537#
$FFLG   063460          12500#  12503#  12531  12540*  12548#
$FILLC  001172           1343#  11920  11951
$FILLS  001171           1342#  11951
$GDADR  001134           1326#  4474*  4489   4989*  5004   8757*  12555
$GDDAT  001140           1328#  4466*  4467*  4487*  4488*  4493   4982*  4983*  5002*  5003*  5009   6959*  6960*
                         7013#  7014#  7151#  7152#  7289#  7290#  7440#  7441#  7456#  7457#  7475#  7476#  7672#
                         7673#  8755#  9072#  9073#  9074   9090#  9091#  9097#  9099#  9111#  9112#  9113#  9129#
                         9130#  9145#  9146#  9173#  9174#  9227#  9240#  9251#  9252   9264#  9269#  9272   9283#
                         9291#  9294   9320#  9323#  9324#  9327   9339#  9340#  9343   9350#  9702#  9708#  9709#
                         9710#  9711#  9714#  9715   9762#  9767#  9770   9779#  9792#  9797   9808#  9809#  9810#
                         9812   9821#  9822#  9824   9940#  9952#  9965#  9970   9989#  10003#  10004#  10031#  10044#
                        10068#  10069#  10083#  10096#  10097#  10112#  10113#  10200#  10201  10210#  10226#  10228  10237#
                        10250#  10251  10260#  10272#  10273  10282#  10295#  10296  10336#  10337#  10350#  10351#  10369#
                        10370#  10383#  10384#  10397#  10398#  10411#  10412#  10425#  10426#  10477#  10478#  10491#  10492#
                        10507#  10508#  10524#  10525#  10545#  10546#  10562#  10563#  10576#  10577#  10601#  10602#  10617#
                        10619#  10631#  10632#  10647#  10648#  10664#  10665#  10682#  10683#  10696#  10697#  10712#  10713#
                        10749#  10750  10761#  10762  10771#  10802#  10803  10814#  10815  10823#  10882#  10883#  10898#
                        10899#  10919#  10920#  10939#  10940#  10968#  10969#  10987#  10988#  11001#  11002#  11015#  11016#
                        11026#  11039#  11042  11056#  11057#  11089#  11090#  11105#  11106#  11119#  11120#  11137#  11138#
                        11150#  11151#  11165#  11166#  11174#  11205#  11206#  11218#  11219#  11231#  11232#  11244#  11245#
```

```
                    11266* 11267* 11291* 11292* 11306* 11307* 11319* 11320* 11332* 11333* 11345* 11346* 11366*
                    11367* 11383* 11384* 11399* 11400* 11464* 11465* 11480* 11496* 11497* 11516* 11517* 11530*
                    11531* 11589* 11604* 12555
$GET42  033146       7780*
$GTSWR  061756      12216* 12446
$HD   = 000001        812    813
$HIBTS  001100       1301*
$HIOCT  062726      12398* 12403*
$ICNT   001120       1319* 12011* 12012  12014* 12025
$ILLUP  063176      12458  12474  12491*
$INTAG  001151       1333* 12213* 12232  12252  12365
$ITEMB  001130       1323*  7829  12081* 12089  12110
$LF     001220       1355* 11951  12110  12348  12358
$LFLG   063457      12541* 12547*
$LPADR  001122       1320*  3249*  3525* 11993* 12002* 12018* 12023  12025
$LPERR  001124       1321*  3250*  3526* 12002  12019* 12025  12100
$MADR1  001254       1388*
$MADR2  001260       1392*
$MADR3  001264       1395*
$MADR4  001270       1398*
$MAIL   001222       1302   1306  1361*  3267  3282  3503  3531  3571  3615  3653  3821  3989  4159
                     4319   4507  4675  4835  5023  5182  5341  5500  5665  5810  5955  6100  6185
                     6267   6414  6561  6707  6882  7035  7173  7311  7547 11895 12017 12087
$MAMS1  001252       1382*
$MAMS2  001256       1390*
$MAMS3  001262       1393*
$MAMS4  001266       1396*
$MBADR  001102       1302*
$MFLG   063456      12501* 12507  12542* 12546*
$MNEW   062614      12219  12363*
$MSGAD  001236       1368* 12517* 12520
$MSGLG  001240       1369* 12522*
$MSGTY  001222       1362* 12515  12523* 12535  12539*
$MSWR   062603      12216  12361*
$MTYP1  001253       1383*
$MTYP2  001257       1391*
$MTYP3  001263       1394*
$MTYP4  001267       1397*
$MXCNT  061074      12015  12025*
$NULL   001170       1341* 11922  11951
$NWTST= 000001       3520*  3522  3561*  3563  3605*  3607  3644*  3812*  3980*  4150*  4310*  4498*  4666*
                     4826*  5014*  5173*  5332*  5491*  5656*  5801*  5946*  6091*  6176*  6258*  6405*  6552*
                     6698*  6873*  7026*  7164*  7302*  7538*
$OCNT   060272      11827* 11856* 11869*
$OMODE  060274      11822* 11826* 11831  11834* 11845* 11871*
$OVER   061060      11969  11994  12003  12013  12022*
$PASS   001230       1365*  3267*  7756*  7757*  7768  . 7790  12009  12026
$PASTM  001106       1304*
$POWER  063204      12489  12494*
$PWRDN  063036       3243  12458* 12486
$PWRMG  063172      12489*
$PWRUP  063110      12468  12474*
$QUES   001216       1353* 11951  12110  12270  12341  12358
$RDCHR  062230      12283* 12449
$RDDEC= ******      12452
$RDLIN  062320      12306* 12450
```

```
$RDOCT  062626          12376#  12451
$RDSZ = 000010          12299#
$RESRE  057512          11692#  12453
$RTNAD  033170           7789#
$R2A  = ****** U        12454
$SAVRE  057454          11676#  12452
$SAVR6  063202          12467#  12475   12476#  12477#  12493#
$SCOPE  060560           3237   11965#
$SETUP= 000137           1255#   3236    3237    3239    3241    3243    3245    3246    3247    3249    3276    3279    3500
                         7754   11966   12071   12097   12105   12153   12158   12159   12189   12365
$STUP = 177777           1255#
$SVLAD  061024          11977   12016#
$SVPC = 000200           1271#   1276
$SWR  = 167400            801#    812     817     818     819     820     821     822     823     824    1350    1351    1352
                         3246    3247    3249    3250    3531    3571    3615    3653    3821    3989    4159    4319    4507
                         4675    4835    5023    5182    5341    5500    5665    5810    5955    6100    6185    6267    6414
                         6561    6707    6882    7035    7173    7311    7547    7749    7755    7782    7788    7790   11957
                        11958   11959   11960   11961   11968   11980   11982   11983   11996   11997   11998   12005   12006
                        12007   12019   12022   12025   12062   12063   12064   12065   12066   12075   12082   12094   12098
                        12110   12490
$SWREG  001244           1373#   3270
$SWRMK= 000000            824     825   11961   11962   11986
$SWO8T  061076          11992   12026#
$TESTN  001226           1364#   3531#   3571#   3615#   3653#   3821#   3989#   4159#   4319#   4507#   4675#   4835#   5023#
                         5182#   5341#   5500#   5665#   5810#   5955#   6100#   6185#   6267#   6414#   6561#   6707#   6882#
                         7035#   7173#   7311#   7547#   7821  12017#
$TIMES  001206           1350#   3246#   7755#  12005#  12012  12015#  12025
$TKB    001162           1338#  12113   12134   12145   12170   12198   12225
$TKCNT  061370          12114#  12129#  12159   12176#  12290   12292#
$TKINT  061400           3340    3513   12129#  12150   12211
$TKQEN= 061377          12118#  12184   12295
$TKQIN  061372          12115#  12130#  12131   12182#  12183#  12184   12186#
$TKQOU  061374          12116#  12131#  12293   12294#  12295   12297#
$TKQSR  061376          12117#  12130   12186   12297
$TKS    001160           1337#  12113   12135#  12166#  12168   12174#  12196   12212#  12222   12234#  12254#
$TKSRV  061450          12132   12145#
$TMP0   001174           1345#   3585#   3588    3663#   3831#   3993#   4169#   4329#   4517#   4685#   4845#   5033#   5192#
                         5351#   5510#   5675#   5820#   5965#   6277#   6424#   6571#   6717#   6892#   7045#   7183#   7322#
                         7458#   7477#   7558#   7674#   8652    8759#   8848#   8921#   9876#   9877   12555
$TMP1   001176           1346#   3350#   3351    3352    3355    3364#   3365    3366    3374#   3375    3378    3388#   3390
                         3394    3395#   3396#   3398    3404#   3406    3410    3411#   3412#   3413#   3414#   3415#   3416#
                         3417#   3419    3425#   3427    3432#   3433#   3434#   3435#   3436#   3437    3445#   3446    3447
                         3454#   3455    3456    3458    3460    3464    3584#   3592    3638#   3664#   3832#   4000#   4170#
                         4330#   4518#   4686#   4846#   5034#   5193#   5352#   5511#   5676#   5821#   5966#   6278#   6425#
                         6572#   6718#   6893#   7046#   7184#   7323#   7459#   7478#   7559#   7675#   8653    8849#   8850
                         8922#   8923    9875#   9882   11573#  11574#  11586   11588   12555
$TMP2   001200           1347#   3639#   7460#   7479#   7676#  11576#  11577#  11600   11603   12555
$TMP3   001202           1348#
$TMP4   001204           1349#
$TN   = 000036            812#   3520    3531#   3561    3571#   3605    3615#   3644    3653#   3812    3821#   3980    3989#
                         4150    4159#   4310    4319#   4498    4507#   4666    4675#   4826    4835#   5014    5023#   5173
                         5182#   5332    5341#   5491    5500#   5656    5665#   5801    5810#   5946    5955#   6091    6100#
                         6176    6185#   6258    6267#   6405    6414#   6552    6561#   6698    6707#   6873    6882#   7026
                         7035#   7164    7173#   7302    7311#   7538    7547#  11987   12027
$TPB    001166           1340#  11940#  11951
$TPFLG  001173           1344#  11889   11951
```

```
$TPS     001164          1339#  11938   11951
$TRAP    062730          3241   12412#
$TRAP2   062770         12427#  12438
$TRP   = 000016         12431#  12440#  12441#  12442#  12443#  12444#  12445#  12446   12447#  12448   12449#  12450#  12451#
                        12452#  12453#  12454#
$TRPAD   063002         12421   12438#
$TSTM    001104          1303#
$TSTNM   001116          1317#   7754#  11956   11989#  12016#  12017   12022   12026   12074   12110
$TTYIN   062554         12308   12309   12321   12339   12353   12357#
$TYPBN   057550         11713#  12444
$TYPDS   057624         11740#  12443
$TYPE    060276         11889#  12431   12439   12528
$TYPEC   060510         11919   11926   11933   11938#  11939   12256
$TYPEX   060556         11944   11946   11949#
$TYPOC   060074         11825#  12440
$TYPON   0601.0         11824   11827#  12442
$TYPOS   060050         11820#  12441
$UNIT    001234          1367#   3518#   7815    9072
$UNITM   001110          1305#
$USWR    001246          1374#
$VECT1   001272          1399#   3395    3410#   3411    3437#
$VECT2   001274          1400#
$XTSTR   060576         11971#
$$GET4=  000000          7782#
$$SW08=  000036         12026#  12027   12028#  12029#  12030#  12031#  12032#  12033#  12034#  12035#  12036#  12037#  12038#
                        12039#  12040#  12041#  12042#  12043#  12044#  12045#  12046#  12047#  12048#  12049#  12050#  12051#
                        12052#  12053#  12054#  12055#  12056#
$OFILL   060273         11821#  11825#  11835   11870#
$OCAT=   ****** U       11968   12084
.      = 106121          1258#   1262#   1271    1272#   1274#   1276#   1281#   1284#   1290    1291#   1293#   1295#   1307
                         1314#   1356    1475#   1491#   1496#   3234    3249    3250    7767#   7790    7791#   11644   11656#
                        11794#  11951   12025   12026   12110   12113   12117#  12118   12119#  12357#  12358   12365#  12470
                        12492   12549#  12555#
.$$STA=  ****** -       12501   12504
 $      = 001100          1290#   1295
```

```
CALCLR    801#  3573  3617
CALGET    801#  3698  3730  3773  3866  3898  3941  4034  4067  4111  4204  4236  4278  4364  4396
         4441  4481  4552  4584  4627  4720  4752  4794  4880  4912  4957  4996  5068  5100  5142
         5227  5259  5301  5386  5418  5460  5547  5580  5622  5711  5754  5856  5899  6001  6044
         6144  6226  6313  6367  6460  6514  6607  6661  6765  6827  6941  6995  7081  7125  7219
         7263  7367  7410  7603  7646
CALPUT    801#  3685  3721  3764  3853  3889  3932  4021  4058  4102  4191  4227  4269  4351  4387
         4432  4539  4575  4618  4707  4743  4785  4867  4903  4948  5055  5091  5133  5214  5250
         5292  5373  5409  5451  5534  5571  5613  5699  5745  5844  5890  5989  6035  6133  6214
         6301  6355  6448  6502  6595  6649  6739  6753  6802  6818  6910  6929  6983  7069  7116
         7207  7254  7355  7401  7528  7591  7637  7723
CALSJB    801#  3669  3706  3738  3745  3752  3781  3788  3795  3804  3837  3874  3906  3913  3920
         3949  3956  3963  3972  4005  4042  4075  4082  4089  4119  4126  4133  4142  4175  4212
         4244  4251  4258  4286  4293  4300  4335  4372  4404  4411  4418  4449  4460  4523  4560
         4592  4599  4606  4635  4642  4649  4658  4691  4728  4760  4767  4774  4802  4809  4816
         4851  4888  4920  4927  4934  4965  4976  5039  5076  5108  5115  5122  5150  5157  5164
         5198  5235  5267  5274  5281  5309  5316  5323  5357  5394  5426  5433  5440  5468  5475
         5482  5516  5555  5588  5595  5602  5630  5637  5644  5681  5719  5726  5733  5762  5769
         5776  5785  5826  5864  5871  5878  5907  5914  5921  5930  5971  6009  6016  6023  6052
         6059  6066  6075  6104  6152  6159  6166  6188  6234  6241  6248  6283  6321  6328  6335
         6344  6375  6382  6389  6430  6468  6475  6482  6491  6522  6529  6536  6577  6615  6622
         6629  6638  6669  6676  6683  6723  6773  6780  6787  6835  6842  6849  6858  6898  6949
         6963  6972  7003  7017  7051  7089  7096  7103  7133  7144  7155  7189  7227  7234  7241
         7271  7282  7293  7328  7375  7382  7389  7418  7433  7448  7468  7486  7513  7564  7611
         7618  7625  7654  7665  7683  7708
CNACMS   3520#  3522
COMMEN    943#
DVAVMS   3561#  3563
DVTYMS   3605#  3607
ENDCOM    943#
ERROR     837#  3496  3554  3576  3590  3594  3600  3620  3629  3640  3673  3688  3701  3709  3724
         3733  3741  3748  3755  3767  3776  3784  3791  3798  3809  3841  3856  3869  3877  3892
         3901  3909  3916  3923  3935  3944  3952  3959  3966  3977  4009  4024  4037  4045  4061
         4070  4078  4085  4092  4105  4114  4122  4129  4136  4147  4179  4194  4207  4215  4230
         4239  4247  4254  4261  4272  4281  4289  4296  4303  4339  4354  4367  4375  4390  4399
         4407  4414  4421  4435  4444  4452  4463  4469  4484  4491  4495  4527  4542  4555  4563
         4578  4587  4595  4602  4609  4621  4630  4638  4645  4652  4663  4695  4710  4723  4731
         4746  4755  4763  4770  4777  4788  4797  4805  4812  4819  4855  4870  4883  4891  4906
         4915  4923  4930  4937  4951  4960  4968  4979  4985  4999  5006  5011  5043  5058  5071
         5079  5094  5103  5111  5118  5125  5136  5145  5153  5160  5167  5202  5217  5230  5238
         5253  5262  5270  5277  5284  5295  5304  5312  5319  5326  5361  5376  5389  5397  5412
         5421  5429  5436  5443  5454  5463  5471  5478  5485  5520  5537  5550  5558  5574  5583
         5591  5598  5605  5616  5625  5633  5640  5647  5685  5702  5714  5722  5729  5736  5748
         5757  5765  5772  5779  5790  5830  5847  5859  5867  5874  5881  5893  5902  5910  5917
         5924  5935  5975  5992  6004  6012  6019  6026  6038  6047  6055  6062  6069  6080  6108
         6136  6147  6155  6162  6169  6192  6217  6229  6237  6244  6251  6287  6304  6316  6324
         6331  6338  6348  6358  6370  6378  6385  6392  6434  6451  6463  6471  6478  6485  6495
         6505  6517  6525  6532  6539  6581  6598  6610  6618  6625  6632  6642  6652  6664  6672
         6679  6686  6727  6742  6756  6768  6776  6783  6790  6805  6821  6830  6838  6845  6852
         6863  6902  6913  6932  6944  6952  6961  6966  6976  6986  6998  7006  7015  7020  7055
         7072  7084  7092  7099  7106  7119  7128  7136  7147  7153  7158  7193  7210  7222  7230
         7237  7244  7257  7266  7274  7285  7291  7296  7332  7358  7370  7378  7385  7392  7404
         7413  7421  7436  7442  7451  7471  7480  7489  7517  7531  7568  7594  7606  7614  7621
         7628  7640  7649  7657  7668  7677  7686  7712  7726
ESCAPE    943#
GETCSI    801#
```

```
GETDB      801#
GETDS      801#
GETDT      801#    3623
GETMR1     801#
GETPR1     943#
GETSWR     801#     943#    3279#    3500
MULT       943#
NEWTST     943#    3520    3561    3605    3644    3812    3980    4150    4310    4498    4666    4826    5014    5173    5332
          5491    5656    5801    5946    6091    6176    6258    6405    6552    6698    6873    7026    7164    7302    7538
NWTST      801#    3523    3565    3609    3647    3815    3983    4153    4313    4501    4669    4829    5017    5176    5335
          5494    5659    5804    5949    6094    6179    6261    6408    6555    6701    6876    7029    7167    7305    7541
POP        943#    3547    3548    3552    3553    3586    3587    3598    3599    8213    8432    8433    8446    8589    8590
          8591    8663    8665    8666    8667    8668    8769    8771    8772    8773    8802    8803    8804    8877    8879
          8880    8881    8882    8883    8884    8944    8946    8947    8948    8949    8950    8951    8973    8975    9015
          9017    9018    9019    9020    9168    9172    9316    9900    9901    9902    9903   10163   10165   10166   10167
         10227   10789   10790   11697   11781   12399   12479   12480   12543   12544
PUSH       943#    3533    3534    3578    3580    8190    8415    8416    8439    8457    8458    8459    8615    8617    8618
          8619    8620    8691    8692    8693    8694    8786    8788    8789    8836    8837    8838    8839    8840    8841
          8842    8909    8910    8911    8912    8913    8914    8915    8957    8958    8986    8987    8988    8989    8990
          9164    9313    9864    9865    9866    9867   10147   10149   10150   10151   10221   10782   10783   11677   11740
         12378   12460   12466   12504   12506   12527
PUTCS1     801#
PUTDC      801#
PUTER1     801#
PUTMR1     801#
REPORT     943#
RGBFMC     801#    1421    1448
SCOPE      808#    3566    3610    3648    3816    3984    4154    4314    4502    4670    4830    5018    5177    5336    5495
          5660    5805    5950    6095    6180    6262    6409    6556    6702    6877    7030    7168    7306    7542
SETPRI     943#    3224    3514   11628   11634   12286
SETTRA   12431#   12440   12441   12442   12443   12444   12446   12448   12449   12450   12451   12452   12453
SETUP      943#    3229
SKIP       943#
SLASH      943#
SPACE      943#
STARS      943#    1269    1287    1289    1296    1310    1356    1359    3520    3522    3561    3564    3605    3608    3644
          3646    3812    3814    3980    3982    4150    4152    4310    4312    4498    4500    4666    4668    4826    4828
          5014    5016    5173    5175    5332    5334    5491    5493    5656    5658    5801    5803    5946    5948    6091
          6093    6176    6178    6258    6260    6405    6407    6552    6554    6698    6700    6873    6875    7026    7028
          7164    7166    7302    7304    7538    7540    7745    7794    7995    8008    8020    8042    8071    8093    8121
          8185    8451    9211    9219    9304    9687   11661   11706   11730   11797   11874   11953   12058   12112   12189
         12204   12275   12299   12368   12406   12456   12472   12499
SWRSU      943#    3251#
TAGS       801#    1415
TRMTRP   12431#
TYPBIN     943#
TYPDEC     943#    7768    7775
TYPNAM     801#     943#    3272
TYPNUM     943#
TYPOCS     943#    3398    3419    7815    7823    7832    7838
TYPOCT     943#    3384   12217
TYPTXT     943#    7764    7771
XPEP       801#    1518    1525    1532    1539    1546    1553    1560    1567    1574    1581    1588    1595    1602    1609
          1616    1623    1630    1637    1644    1651    1658    1665    1672    1679    1686    1693    1700    1707    1714
          1721    1728    1735    1742    1749    1756    1763    1770    1777    1784    1791    1798    1805    1813    1820
          1827    1834    1841    1849    1857    1865    1872    1879    1886    1893    1900    1907    1914    1921    1929
```

```
                1936    1943    1950    1957    1964    1971    1978    1985    1992    1999    2006    2013    2020    2027    2034
                2041    2048    2055    2062    2069    2076    2083    2090    2097    2104    2111    2118    2125    2132    2139
                2146    2153    2160    2167    2174    2181    2188    2195    2202    2209    2216    2223    2230    2237    2244
                2251    2258    2265    2272    2279    2286    2293    2300    2307    2314    2322    2329    2336    2343    2350
                2357    2364    2371    2378    2385    2392    2399    2406    2413    2420    2427    2434    2441    2448    2455
                2462    2469    2476    2483    2490    2497    2504    2511    2518    2525    2532    2539    2546    2553    2560
                2567    2574    2581    2588    2595    2602    2609    2616    2623    2630    2637    2644    2651    2658    2665
                2672    2679    2686    2693    2700    2707    2714    2721    2728    2735    2742    2749    2756    2763    2770
                2777    2784    2791    2799    2806    2814    2821    2828    2836    2844    2852    2861    2869    2877    2884
                2891    2898    2905    2912    2919    2926    2933    2940    2947    2954    2961    2968    2975    2982    2989
                2996    3003    3010    3017    3024    3031    3038    3045    3052    3059    3066    3073    3080    3087    3094
                3101    3108    3115    3122    3129    3136    3143    3150    3157    3164    3171    3178
$$CMRE          1308#
$$CMTM          1308#   1345    1346    1347    1348    1349
$$ESCA           943#
$$NEWT           943#   3520    3561    3605    3644    3812    3980    4150    4310    4498    4666    4826    5014    5173    5332
                5491    5656    5801    5946    6091    6176    6258    6405    6552    6698    6873    7026    7164    7302    7538
$$SET          12431#  12440   12441   12442   12443   12444   12446   12448   12449   12450   12451   12452   12453
$$SETM          3267#
$$SKIP           943#
.EQUAT           801#    833
.HEADE           801#    802
.SETUP           801#   1255
.SWRHI           801#    813
.SWRLO           801#    825#
.$ACTI           801#   1267
.$APTB           801#   1357#
.$APTH           801#   1285
.$APTY           801#  12497
.$CATC           801#   1256
.$CMTA           801#   1308
.$DIV            801#
.$EOP            801#   7743
.$ERRO           801#  12056
.$ERRT           801#
.$MULT           801#
.$POWE           801#  12454
.$RAND           801#
.$RODE           801#
.$RDOC           801#  12366
.$READ           801#  12110
.$SAVE           801#  11659
.$SCOP           801#  11951
.$SIZE           801#
.$TRAP           801#  12404
.$TYPB           801#  11704
.$TYPD           801#  11728
.$TYPE           801#  11872
.$TYPO           801#  11795

. ABS.   106121      000

 ERRORS DETECTED:  0
```

```
RM03:CZRMD8.BIN,RM03:CZRMD8.SEQ/DOC/NL:TOC/SOL/CRF=RM03:CZRMD8.P11
RUN-TIME:  65 61 4 SECONDS
RUN-TIME RATIO:  599/130=4.6
CORE USED:  33K  (65 PAGES)

DOCUMENT PAGES:  311
```