RM03

FUNCTIONAL TEST PART 1

MD-11-DZRMC-A

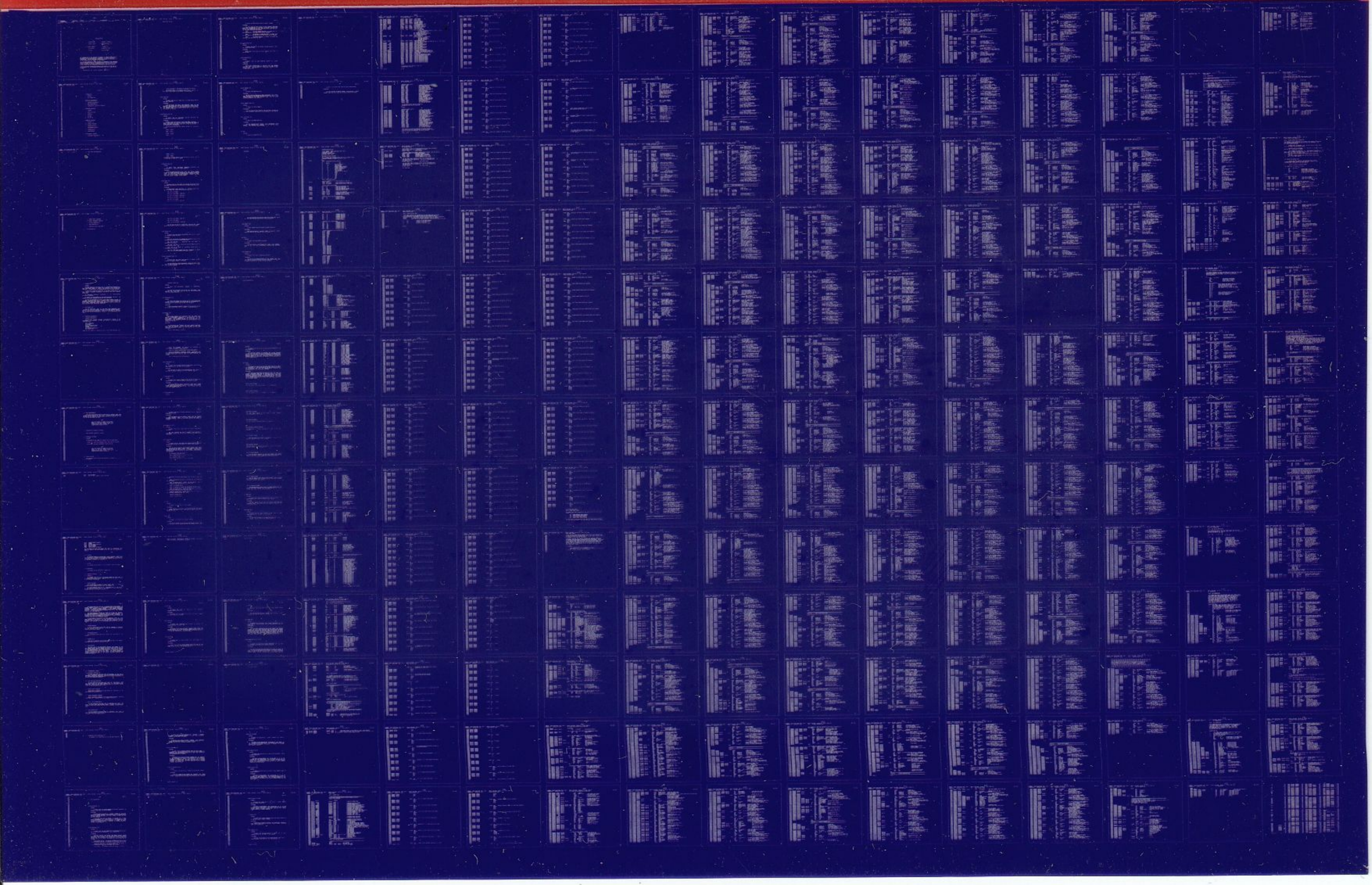
EP-DZRMC-A-DL-A

FICHE 1 OF 2

OCT 1977

digital

MADE IN USA



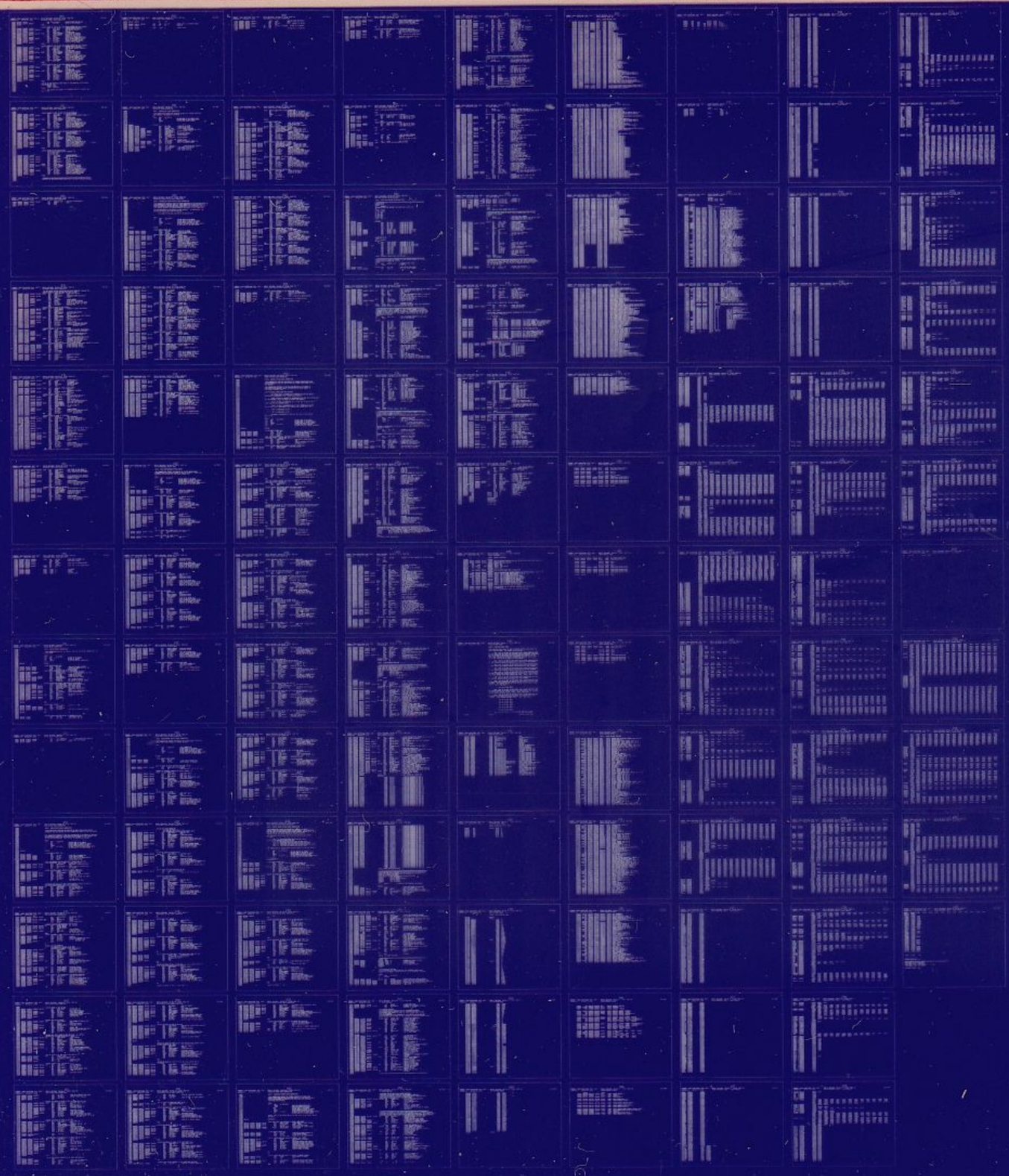
RM03

FUNCTIONAL TEST PART 1
MD-11-DZRMC-A

EP-DZRMC-A-DL-A

FICHE 2 OF 2

OCT 1977
digital
MADE IN USA



```
1                               .REM    \
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
```

IDENTIFICATION

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

```
                        .PAGE
```

CONTENTS

109 2. DUAL PORT CONFIGURATIONS

3. MEMORY PARITY HARDWARE

4. MEMORY MANAGEMENT HARDWARE

5. ACT,APT COMPATIBILITY

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

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

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

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

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

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

1.2 UNIT UNDER TEST

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

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

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

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

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

```
   182                                  DRIVES AND DISK PACKS.
```

2.2 MEDIA REQUIREMENTS

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

NOTE

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

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, DZRMJ-A

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

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

3.2 SWITCH OPTIONS

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

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

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

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

3.3 STARTING

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

3.4 HALTING

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

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

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

4.2 CONSOLE DIALOGUE

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

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

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

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

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

4.3 PROGRESS REPORTS

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

4.4 PERFORMANCE REPORT

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

4.5 PROGRAM HALTS

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

4.6 ERROR REPORTS

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

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

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

5.2 DUAL PORT CONFIGURATIONS

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

5.3 MEMORY PARITY HARDWARE

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

5.4 MEMORY MANAGEMENT HARDWARE

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

5.5 ACT11, APT11 COMPATIBILITY

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

5.6 XXDP COMPATIBILITY

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

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

CONTROLLER ACCESS TEST

PURPOSE:

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

PROCEDURE:

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

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

DEVICE AVAILABLE TEST

PURPOSE:

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

PROCEDURE:

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

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

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

.BUS TIMEOUT - THE MASSBUS CONTROLLER FAILED TO DETECT A

```
   466                                  NONEXISTENT DEVICE;
```

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

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

DRIVE TYPE TEST

PURPOSE:

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

PROCEDURE:

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

UNIBUS INITIALIZE TEST

PURPOSE:

TO VERIFY THAT ALL APPLICABLE SUBSYSTEM REGISTERS ARE INITIALIZED BY THE RESET INSTRUCTION.

PROCEDURE:

NONZERO VALUES ARE WRITTEN IN EACH APPLICABLE REGISTER. A RESET INSTRUCTION IS EXECUTED, AND THE REGISTERS ARE TESTED TO INSURE THEY WERE INITIALIZED. THIS TEST IS DONE ONCE BECAUSE OF APT COMPATIBILITY REQUIREMENTS.

THE FOLLOWING REGISTERS ARE PRESET BEFORE THE INITIALIZE OCCURS:

RMCS1 - 003577

RMBA - 777776

RMCS2 - 021037

RMER1 - 777777

RMER3 - 777777

RMMR - 040001

IN ADDITION, THE DATA BUFFER IS USED TO FORCE DLT,TRE,SC AND OR TO A ONE AND TO FORCE IR TO A ZERO.

CONTROLLER CLEAR TEST

PURPOSE:

TO VERIFY THAT APPLICABLE SUBSYSTEM REGISTERS ARE INITIALIZED BY A "CONTROLLER CLEAR" OPERATION.

PROCEDURE:

LIKE THE UNIBUS INITIALIZE TEST, THIS TEST WRITES NONZERO VALUES IN THOSE REGISTERS WHICH ARE INITIALIZED BY CONTROLLER CLEAR. THE SUBSYSTEM IS THEN CLEARED USING A CONTROLLER CLEAR, I.E., BIT 5 OF CONTROL STATUS REGISTER 2 (RMCS2), AND EACH REGISTER IS READ TO INSURE IT WAS CLEARED.

ERROR CLEAR TEST

PURPOSE:

TO VERIFY THAT ALL APPLICABLE RH70 MASSBUS CONTROLLER STATUS AND ERROR CONDITIONS ARE INITIALIZED BY AN ERROR CLEAR OPERATION.

PROCEDURE:

AN "RH70 ERROR CLEAR" OPERATION, I.E., WRITING A ONE IN TRE, BIT 14 OF RMCS1 WILL CLEAR THE FOLLOWING STATUS BITS:

.TRE, BIT 14 OF RMCS1

.MCPE, BIT 13 OF RMCS1 - READ ONLY

.DLT, BIT 15 OF RMCS2 - READ ONLY

.WCE, BIT 14 OF RMCS2 - READ ONLY

.UPE, BIT 13 OF RMCS2

.NED, BIT 12 OF RMCS2 - READ ONLY

.PGE, BIT 10 OF RMCS2 - READ ONLY

.MXF, BIT 09 OF RMCS2

.MDPE, BIT 08 OF RMCS2 - READ ONLY

THE TEST SETS UPE AND MXF STATUS BITS, THEN SETS TRE (ERROR CLEAR) AND VERIFIES THAT ALL THE ABOVE STATUS BITS ARE CLEARED.

DRIVE STATUS TEST

PURPOSE:

TO VERIFY THAT THE STORAGE MODULE DISK DRIVE IS IN A STATE THAT PERMITS FURTHER TESTING.

PROCEDURE:

THIS TEST INITIALIZES THE MASSBUS AND EXAMINES STATUS OF THE SELECTED DEVICE FOR THE FOLLOWING CONDITIONS:

.MOL, BIT 12 OF RMDS =1, INDICATING THAT UNIT READY IS ASSERTED BY THE DRIVE;

.WRL, BIT 11 OF RMDS =0, INDICATING THAT THE DRIVE IS NOT IN A WRITE PROTECT STATE;

.DVC, BIT 07 OF RMER3 =0, INDICATING DRIVE FAULT IS UNASSERTED BY THE DRIVE;

.UNS, BIT 14 OF RMER1 SHOULD EQUAL DVC, OTHERWISE AC POWER IS LOW OR A FAILURE HAS OCCURRED WITH UNSAFE STATUS.

PRIMARY/SECONDARY ERROR TEST

PURPOSE:

TO VERIFY THAT THE RM03 CAN EXECUTE A COMMAND WITHOUT INCURRING UNEXPECTED ERRORS. PROCEDURE:

THE TEST EXECUTES A PACK ACKNOWLEDGE COMMAND AND MAKES SURE THAT GO RESETS AND THAT THERE ARE NO PARITY ERRORS, ETC. VOLUME VALID STATUS IS IGNORED.

DIAGNOSTIC MODE TEST

PURPOSE:

TO VERIFY THAT MAINTENANCE HARDWARE IS OPERATIONAL.

PROCEDURE:

THE TEST THAT DIAGNOSTIC MODE CAN BE SET AND RESET, THEN VERIFIES THAT "MOL, PIP, WRL, SKI, AND DVC" CAN BE CONTROLLED USING MAINTENANCE REGISTER'1.

PACK ACKNOWLEDGE TEST

PURPOSE:

TO VERIFY THAT VOLUME VALID CAN BE SET BY A PACK ACKNOWLEDGE COMMAND, LENDING CREDENCE TO THE EXECUTION OF THE COMMAND AND TO THE STABILITY OF THE UNIT READY SIGNAL FROM THE DRIVE.

PROCEDURE:

A PACK ACKNOWLEDGE COMMAND IS ISSUED TO THE SELECTED DEVICE AND VOLUME VALID STATUS, BIT 10 OF RMDS, IS CHECKED FOR ONE.

RECALIBRATE TEST

PURPOSE:

THE PRIMARY PURPOSE IS TO ASCERTAIN THAT THE DRIVE WILL EXECUTE A RECALIBRATE OPERATION TO THE EXTENT THAT "PIP" AND "SKI" STATUS BECOME UNASSERTED AT THE COMPLETION OF THE RECALIBRATE. THE SECONDARY PURPOSE IS TO PUT THE DRIVE IN A KNOWN STATE SO THAT FURTHER TESTS CAN CHECK FOR UNEXPECTED STATE CHANGES IN THE DRIVE.

PROCEDURE:

THE RECALIBRATE TEST PRESETS THE DISK ADDRESS REGISTER, RMDA, AND THE DESIRED CYLINDER REGISTER, RMDC, TO ZERO, THEN EXECUTES A RECALIBRATE COMMAND. THE TEST VERIFIES THE FOLLOWING CONDITIONS:

.SKI=0, "SEEK ERROR" IS INACTIVE;

.PIP=0, "ON CYLINDER" IS ACTIVE, AS INDICATED BY "POSITIONING IN PROGRESS" BEING INACTIVE;

.IAE=0, NO "INVALID ADDRESS ERROR" DURING RECALIBRATE;

.OPI=0, NO "OPERATION INCOMPLETE ERROR", INDICATING THAT THAT THE DRIVE WAS READY AT THE START OF THE COMMAND AND THAT ON CYLINDER STATUS WENT INACTIVE WHEN THE DRIVE RECEIVED THE COMMAND;

.ATA=1, ATTENTION IS SET BY RECALIBRATE.

ABORT RECALIBRATE TEST

PURPOSE:

TO VERIFY THAT THE RM03 INHIBITS A RECALIBRATE WHEN AN ABORT CONDITION EXISTS AT THE START OF THE COMMAND.

PROCEDURE:

THE TEST SETS AN ERROR IN THE ERROR REGISTER AND ISSUES A RECALIBRATE COMMAND, VERIFYING THAT "PIP" REMAINS INACTIVE.

IVC RECALIBRATE TEST

PURPOSE:

TO VERIFY THAT INVALID COMMAND STATUS, BIT 12 OF RMER2, SETS WHEN VOLUME VALID IS INACTIVE DURING A RECALIBRATE COMMAND.

PROCEDURE:

THE PROGRAM SETS AND RESETS DIAGNOSTIC MODE WHICH CAUSES VOLUME VALID, BIT 6 OF RMDS TO RESET. THE PROGRAM THEN EXECUTES A RECALIBRATE COMMAND AND VERIFIES THAT "IVC" STATUS SETS AND THAT "PIP" REMAINS INACTIVE.

IAE RECALIBRATE TEST

PURPOSE:

TO VERIFY THAT INVALID ADDRESS ERROR DOES NOT SET DURING A RECALIBRATE COMMAND .

PROCEDURE:

THE TEST PRESETS THE DISK ADDRESS (RMDA) AND THE DESIRED CYLINDER ADDRESS (RMDC) TO ILLEGAL VALUES AND ISSUES A RECALIBRATE COMMAND, VERIFYING THAT "IAE" DOES NOT SET DURING THE COMMAND.

RECALIBRATE AT OFFSET

PURPOSE:

TO VERIFY THAT OFFSET MODE DOES NOT AFFECT RECALIBRATE.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND, THEN EXECUTES A RECALIBRATE COMMAND AND VERIFIES THAT THERE ARE NO ERRORS DURING RECALIBRATE.

DRIVE CLEAR TEST

PURPOSE:

TO VERIFY THAT ALL APPLICABLE RM03 MASSBUS ADAPTER ERROR AND STATUS CONDITIONS ARE INITIALIZED BY A "DRIVE CLEAR" COMMAND.

PROCEDURE:

THIS TEST WRITES ONES IN THOSE MASSBUS ADAPTER BITS WHICH ARE INITIALIZED BY DRIVE CLEAR , THEN ISSUES THE DRIVE CLEAR COMMAND AND VERIFIES THAT EACH BIT IS CLEARED. ADDITIONALLY, REGISTERS WHICH ARE NOT AFFECTED BY "DRIVE CLEAR" ARE ALSO PRESET AND VERIFIED.

THE FOLLOWING ITEMS ARE PRESET:

.RMER1, ERROR REGISTER 1 IS SET TO ALL ONES;

.DMD, DIAGNOSTIC MODE IS SET;

.RMER2, ERROR REGISTER 3, IS SET TO ALL ONES.

FOLLOWING THE DRIVE CLEAR COMMAND, THE FOLLOWING ITEMS ARE CHECKED:

.RMCS1, IS CHECKED FOR DVA=1, FD-F4 AND GO=0, ALL OTHER BITS ARE DONT CARES;

.RMDS, IS CHECKED FOR 0, EXCEPT FUR MOL,DPR,DRY, AND VV WHICH SHOULD BE ONE, AND PGM WHICH IS A DONT CARE.

.RMER1, IS CHECKED FOR 0;

.RMAS, IS CHECKED TO INSURE THE APPROPRIATE ATA BIT IS 0;

.RMMR, IS CHECKED FOR 0, EXCEPT FOR WORD CLOCK, LAST SECTOR, AND LAST SECTOR AND TRACK WHICH ARE DONT CARES;

.RMMR2 IS CHECKED FOR 0, EXCEPT FOR THE TEST BIT WHICH IS ONE, AND REQA, REQB WHICH ARE DONT CARES;

.RMEC2 IS CHECKED FOR 0;

.RMER3 IS CHECKED FOR 0;

NOP TEST

PURPOSE:

TO VERIFY THE EXECUTION OF "NOP" COMMAND.

PROCEDURE:

A NOP COMMAND IS EXECUTED ON THE SELECTED DEVICE AND STATUS IS CHECKED TO VERIFY THAT THERE WERE NO ERRORS OR UNEXPECTED CHANGES IN STATUS.

OFFSET TEST

PURPOSE:

TO VERIFY THE EXECUTION OF "OFFSET" COMMAND.

PROCEDURE:

THE OFFSET COMMAND IS EXECUTED AND THE PROGRAM CHECKS THAT OFFSET STATUS, BIT 0 OF RMDS IS SET AND THAT ATTENTION, BIT 15 OF

```
                        RMDS IS ALSO SET.  ADDITIONALLY, CONTROLLER, ADAPTER,  AND  DRIVE
                        STATUS IS CHECKED FOR UNEXPECTED CHANGES.
```

GO/ATA TEST

PURPOSE:

TO VERIFY THAT "ATA" WILL RESET WITH "GO" PROVIDING COMPOSITE ERROR IS INACTIVE.

PROCEDURE:

ATTENTION, BIT 15 OF RMDS, IS SET USING AN OFFSET COMMAND AND RESET USING A NOP COMMAND.

WRITE ATA TEST

PURPOSE:

TC VERIFY THAT ATTENTION CAN BE RESET BY WRITING THE ATTENTION SUMMARY REGISTER, RMAS.

PROCEDURE:

ATTENTION IS SET USING AN OFFSET COMMAND AFTER WHICH THE PROGRAM WRITES A 0 IN THE ATTENTION SUMMARY REGISTER, VERIFYING THT ATTENTION REMAINS SET. FOLLOWING THAT, THE PROGRAM WRITES A 1 IN RMAS AND VERIFIES THAT ATTENTION RESETS.

ERROR/ATA TEST

PURPOSE:

TO VERIFY THAT "GO" DOES NOT RESET "ATA" WHEN THERE IS A COMPOSITE ERROR.

PROCEDURE:

"ATA" IS SET WITH AN OFFSET COMMAND AFTER WHICH ONE OF THE ERROR BITS IS SET. THE PROGRAM THEN ISSUES A NOP COMMAND AND VERIFIES THAT THE ATTENTION BIT REMAINS SET.

PROGRAM INTERRUPT TEST

```
915                                     PURPOSE:
```

TO VERIFY THAT THE RM03 SUBSYSTEM WILL GENERATE A PROGRAM INTERRUPT WHEN PROCESSOR PRIORITY IS LESS THAN CONTROLLER PRIORITY AND INTERRUPT IS ENABLED.

PROCEDURE:

ATTENTION IS SET USING AN OFFSET COMMAND. WITH INTERRUPT ENABLED AND PROCESSOR PRIORITY SET BELOW CONTROLLER PRIORITY, THE PROGRAM VERIFIES THAT A PROGRAM INTERRUPT OCCURS.

INHIBIT INTERRUPT TEST

PURPOSE:

TO VERIFY THAT A PROGRAM INTERRUPT DOES NOT OCCUR WHEN (1) PROCESSOR AND CONTROLLER PRIORITY ARE THE SAME AND INTERRUPT IS ENABLED OR (2) WHEN PROCESSOR PRIORITY IS LESS THAN CONTROLLER AND INTERRUPTS ARE NOT ENABLED.

PROCEDURE:

ATTENTION IS SET USING AN OFFSET COMMAND WITH THE PRIORITY OF THE PROCESSOR SET EQUAL TO THE PRIORITY OF THE CONTROLLER. INTERRUPT IS ENABLED AND THE TEST VERIFIES THAT A PROGRAM INTERRUPT DOES NOT OCCUR. INTERRUPTS ARE DISABLED AND PROCESSOR PRIORITY IS LOWERED AND THE TEST VERIFIES THAT A PROGRAM INTERRUPT DOES NOT OCCUR.

RETURN TO CENTERLINE TEST

PURPOSE:

TO VERIFY THE EXECUTION OF "RETURN TO CENTERLINE" COMMAND.

PROCEDURE:

THIS TEST ISSUES AN RTC COMMAND AND VERIFIES THAT OFFSET STATUS, BIT 0 OF RMDS IS RESET AND THAT ATTENTION, BIT 15 OF RMDS IS SET. UNEXPECTED STATUS OR ERROR CONDITIONS ARE ALSO VERIFIED.

READ IN PRESET TEST

```
;972
;973                                    PURPOSE:
```

TO VERIFY THE EXECUTION OF "READ IN PRESET" COMMAND.

PROCEDURE:

THIS TEST LOADS NON ZERO VALUES IN THOSE ADAPTER REGISTERS WHICH ARE INITIALIZED BY THE READ IN PRESET COMMAND, THEN EXECUTES THE COMMAND AND VERIFIES THE CONTENTS OF EACH REGISTER. THE FOLLOWING REGISTERS ARE CHECKED:

.RMDA, THE DISK ADDRESS REGISTER, IS LOADED WITH ALL ONES AND VERIFIED TO BE ZERO AFTER THE RIP COMMAND;

.RMDC, THE DESIRED CYLINDER REGISTER, IS LOADED WITH ALL ONES (^O1777) AND VERIFIED TO BE ZERO AFTER THE RIP COMMAND;

.RMOF, THE OFFSET REGISTER, IS PRESET (TO 016000) AND VERIFIED TO BE ZERO AFTER THE TEST;

RMDC CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT WRITING THE DESIRED CYLINDER REGISTER (RMDC) WILL CLEAR OFFSET MODE.

PROCEDURE:

OFFSET MODE IS SET USING THE OFFSET COMMAND, THEN RESET BY WRITING RMDC.

ILLEGAL FUNCTION TEST

PURPOSE:

TO VERIFY THAT THE RM03 SUBSYSTEM DETECTS ALL ILLEGAL FUNCTIONS.

PROCEDURE:

EACH ILLEGAL FUNCTION CODE IS EXECUTED WITH THE PROGRAM VERIFYING THAT "ILLEGAL FUNCTION" STATUS, BIT 0 OF RMER1 IS SET FOR EACH CODE. THE SUBSYSTEM IS INITIALIZED PRIOR TO EACH ILLEGAL FUNCTION TEST.

```
1028
1029
1030
1031                            INVALID COMMAND TEST
1032
1033                                 PURPOSE:
1034
1035                                      TO VERIFY IVC ERROR DETECTION.
1036
1037                                 PROCEDURE:
1038
1039                                      THE TEST RESETS VOLUME VALID USING MAINTENANCE UNIT READY,
1040                                 THEN EXECUTES A NOP COMMAND AND VERIFIES THAT IVC IS SET.  THE
1041                                 PROCESS IS REPEATED FOR EACH FUNCTION CODE, WITH IVC BEING
1042                                 CHECKED ACCORDING TO THE FUNCTION.
1043
1044
1045
1046
1047
1048                            INVALID ADDRESS ERROR TEST
1049
1050                                 PURPOSE:
1051
1052                                      TO VERIFY IAE ERROR DETECTION.
1053
1054                                 PROCEDURE:
1055
1056                                      THE TEST EXECUTES EACH FUNCTION CODE WITH RMDA, AND RMDC SET
1057                                 TO ILLEGAL ADDRESSES, AND VERIFIES IAE ACCORDING TO THE FUNCTION.
1058
1059
1060
1061
1062
1063                            WRITE LOCK ERROR TEST
1064
1065                                 PURPOSE:
1066
1067                                      TO VERIFY WLE ERROR DETECTION.
1068
1069                                 PROCEDURE:
1070
1071                                      THE TEST SIMULATES WRITE PROTECT USING MAINTENANCE WRITE
1072                                 PROTECT AND VERIFIES WLE ACCORDING TO THE FUNCTION BEING
1073                                 EXECUTED.  EACH FUNCTION CODE IS TESTED.
1074
1075
1076
1077
1078
1079                            OPI TEST
1080
1081                                 PURPOSE:
1082
1083                                      TO VERIFY OPI ERROR DETECTION.
```

```
1084
1085                                    PROCEDURE:
```

ERROR ABORT TESTS

PURPOSE:

THE TEST EXECUTES EACH FUNCTION CODE WITH A SIMULATED DRIVE OFF LINE CONDITION AND VERIFIES OPI STATUS ACCORDINGLY.

ERROR ABORT TESTS

PURPOSE:

TO TEST COMMAND EXECUTION DURING AN ABORT CONDITION.

PROCEDURE:

EACH FUNCTION CODE IS EXECUTED UNDER A SIMULATED UNSAFE CONDITION AND THE TEST VERIFIES THAT GO IS RESET.

RMR TEST

PURPOSE:

TO VERIFY THAT RMR ERROR IS DETECTED.

PROCEDURE:

THE TEST EXECUTES A NOP COMMAND WITH DEBUG CLOCK ENABLED. WITH GO SET, THE TEST WRITES RMCSI, VERIFYING THAT RMR ERROR SETS.

PARITY ERROR TEST

PURPOSE:

TO VERIFY THAT PARITY ERRORS ARE DETECTED.

PROCEDURE:

WITH PAT SET TO CAUSE BAD PARITY ON THE CONTROL BUS, THE TEST WRITES A SHIFTING ONE BIT PATTERN TO RMDA AND VERIFIES THAT AN ERROR IS DETECTED BY THE RM03 FOR EACH PATTERN.

```
1142
1143                                    ILLEGAL REGISTER TEST
```

PURPOSE:

TO VERIFY THE DETECTION OF ILLEGAL REGISTER ADDRESSES BY THE RM03 SUSBSYSTEM.

PROCEDURE:

EACH REGISTER ADDRESS IS ACCESSED AND ILLEGAL REGISTER STATUS, BIT 1 OF RMER1 IS CHECKED ACCORDING TO THE ADDRESS USED. NOTE THAT THE EXECUTION OF THIS TEST IS DEPENDENT ON THE REGISTER ADDRESS JUMPER IN THE RH70 CONTROLLER BECAUSE THE RANGE OF ADDRESSES WHICH THE CONTROLLER WILL RESPOND TO IS LIMITED BY THE WAY THE JUMPER IS CUT.

SEEK TESTS

PURPOSE:

THE PURPOSE OF EACH OF THE FOLLOWING SEEK TESTS IS TO VERIFY THE EXECUTION OF SEEK OPERATIONS BY THE RM03 SUBSYSTEM USING A SET OF ADDRESSES THAT TEST THE ADAPTER/DEVICE INTERFACE AND ELECTROMECHANICAL HEAD POSITIONING HARDWARE.

PROCEDURE:

EACH TEST WILL RECALIBRATE THE DRIVE IF "PIP" OR "SKI" IS ACTIVE. FOLLOWING THAT, THE TEST EXECUTES A SEEK TO A CYLINDER ADDRESS OR SERIES OF ADDRESSES. AT THE COMPLETION OF EACH SEEK OPERATION, SUBSYSTEM STATUS IS STORED AND THE TEST CHECKS FOR PRIMARY ERRORS WHICH PRECLUDE FURTHER ERROR CHECKING. IF THERE ARE NO PRIMARY ERRORS THE TEST CHECKS FOR OPERATIONAL ERRORS AND THEN SECONDARY ERRORS.

SEEK TO LAST CYLINDER

THIS TEST SEEKS TO THE LAST CYLINDER, I.E., CYLINDER 822.

SEEK TO FIRST CYLINDER

THIS TEST SEEKS TO THE FIRST CYLINDER, I.E. CYLINDER 0.

```
1199
1200
1201
1202                    SEEK PRIME CYLINDERS
1203
1204                    THIS  TEST  SEEKS  FORWARD  TO  EACH  PRIME  CYLINDER  ADDRESS,
1205                    I.E.,CYLINDERS 1,2,4,8,...,512.
1206
1207
1208
1209
1210
1211                    SEEK ZERO DIFFERENCE
1212
1213                    THIS TEST EXECUTES SUCCESSIVE SEEKS TO CYLINDER 0.
1214
1215
1216
1217
1218
1219                    SEEK MAXIMUM DIFFERENCE FORWARD
1220
1221                    THIS TEST SEEKS TO CYLINDER 0 FOLLOWED BY  A  SEEK  TO  THE  LAST
1222                    CYLINDER.
1223
1224
1225
1226
1227
1228                    SEEK ADJACENT FORWARD
1229
1230
1231                         THIS TEST SEEKS TO CYLINDER 0, FOLLOWED BY  A  SEEK  TO  THE
1232                    ADJACENT FORWARD CYLINDER, I.E., CYLINDER 1.
1233
1234
1235
1236
1237
1238                    SEEK ADJACENT REVERSE
1239
1240                    THIS TEST SEEKS TO CYLINDER 1, FOLLOWED BY A SEEK TO THE ADJACENT
1241                    REVERSE CYLINDER, I.E., CYLINDER 0.
1242
1243
1244
1245
1246
1247                    SEEK TO INVALID SECTOR
1248
1249                         WITH THE OFFSET REGISTER SET FOR 18 BIT FORMAT, THE  PROGRAM
1250                    SEEKS TO CYLINDER 0, TRACK 0, FOR EACH INVALID SECTOR ADDRESS AND
1251                    VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH SECTOR.
```

SEEK TO INVALID TRACK

THE TEST SEEKS TO EACH INVALID TRACK ADDRESS WITH CYLINDER AND SECTOR ADDRESS SET TO 0 AND VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH TRACK ADDRESS.

SEEK TO INVALID CYLINDER

THE PROGRAM SEEKS TO EACH INVALID CYLINDER ADDRESS WITH THE SECTOR AND TRACK ADDRESS SET TO 0 AND VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH CYLINDER ADDRESSS.

IVC SEEK TEST

PURPOSE:

TO VERIFY THAT INVALID COMMAND STATUS, BIT 12 OF RMER2 SETS WHEN VOLUME VALID IS INACTIVE DURING A SEEK COMMAND.

PROCEDURE:

THE TEST RESETS VOLUME VALID BY SETTING AND RESETTING DIAGNOSTIC MODE, THEN EXECUTES A SEEK COMMAND AND VERIFIES THAT "IVC" STATUS IS SET.

ABORT SEEK TEST

PURPOSE:

TO VERIFY THAT THE RM03 INHIBITS A SEEK WHEN AN ABORT CONDITION EXISTS AT THE START OF THE COMMAND.

PROCEDURE:

THE TEST SETS AN ERROR IN THE ERROR REGISTER AND ISSUES A SEEK COMMAND, VERIFYING THAT "PIP" REMAINS INACTIVE.

```
  1308                                  SEEK AT OFFSET
```

PURPOSE:

TO VERIFY THAT OFFSET MODE DOES NOT CAUSE ERRORS DURING SEEK.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND THEN EXECUTES A SEEK COMMAND, VERIFYING THE RESULTS OF THE SEEK.

LOOK AHEAD TEST

PURPOSE:

TO INSURE THAT THE SECTOR COUNT WHICH ORIGINATES AT THE DRIVE AND IS VISIBLE THROUGH THE LOOK AHEAD REGISTER (RMLA) IS OPERATIONAL.

PROCEDURE:

WITH THE OFFSET REGISTER SET FOR 16 BIT FORMAT, THE PROGRAM SAMPLES THE LOOK AHEAD REGISTER AND COLLECTS EACH DIFFERENT SAMPLE UNTIL THE VALUE OF THE FIRST SAMPLE IS DETECTED OR UNTIL 33 SAMPLES ARE TAKEN. THE COLLECTION IS THEN TESTED TO DETERMINE THAT THE SECTOR COUNT INCREMENTS CORRECTLY THROUGH THE ENTIRE RANGE OF VALID SECTORS. THE SAME PROCEDURE IS REPEATED FOR 18 BIT FORMAT, WHERE THE LIMIT ON THE NUMBER OF SAMPLES IS 31.

SEARCH EACH SECTOR ON CYLINDER

PURPOSE:

TO VERIFY THE EXECUTION OF SEARCH OPERATIONS WITH NO HEAD MOTION USING THE SECTOR PULSE FOR SECTOR COMPARE.

PROCEDURE:

THE TEST INTIALIZES AND RECALIBRATES THE DRIVE IF "PIP" OR "SKI" IS ACTIVE THEN SEEKS TO CYLINDER 0. THE TEST THEN DOES A SEARCH TO EACH SECTOR AND VERIFIES THAT THE SEARCH COMPLETES WITHOUT ERROR. THIS TEST IS DONE ONCE IN 18 BIT FORMAT AND ONCE

```
  1365                                  IN 16 BOT FORMAT.
```

SEARCH OFF CYLINDER

PURPOSE:

TO VERIFY THE EXECUTION OF A SEARCH COMMAND WITH IMPLIED HEAD MOTION.

PROCEDURE:

THE DRIVE IS RECALIBRATED AND INITIALIZED IF THE DRIVE IS OFF CYLINDER OR HAS A SEEK ERROR. THE PROGRAM THEN EXECUTES AND EXPLICIT SEEK TO CYLINDER 0, FOLLOWED BY A SEARCH TO CYLINDER 822, TRACK 0, SECTOR 0.

SEARCH INVALID SECTOR

PURPOSE:

TO VERIFY THE DETECTION OF AN INVALID SECTOR ADDRESS DURING A SEARCH OPERATION.

PROCEDURE:

THE DRIVE IS RECALIBRATED AND INITIALIZED IF IT IS OFF CYLINDER OR HAS A SEEK ERROR. WITH THE OFFSET REGISTER SET FOR 18 BIT FORMAT, THE PROGRAM EXECUTES A SEARCH TO EACH INVALID SECTOR, I.E., SECTORS 30 AND 31 AND VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH SECTOR.

SEARCH INVALID TRACK

PURPOSE:

TO VERIFY THE DETECTION OF AN INVALID TRACK ADDRESS DURING A SEARCH OPERATION.

PROCEDURE:

THE DRIVE IS RECALIBRATED AND INITIALIZED IF IT IS OFF CYLINDER OR HAS A SEEK ERROR. THE PROGRAM EXECUTES A SEARCH TO EACH INVALID TRACK ADDRESS WITH THE CYLINDER ADDRESS 0, AND SECTOR ADDRESS 0 AND VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH TRACK.

SEARCH INVALID CYLINDER

PURPOSE:

TO VERIFY THE DETECTION OF AN INVALID CYLINDER ADDRESS DURING A SEARCH OPERATION.

PROCEDURE:

THE DRIVE IS RECALIBRATED AND INITIALIZED IF IT IS OFF CYLINDER OR HAS A SEEK ERROR. THE PROGRAM THEN EXECUTES A SEARCH TO EACH INVALID CYLINDER ADDRESS AND VERIFIES THAT INVALID ADDRESS ERROR SETS FOR EACH CYLINDER.

IVC SEARCH TEST

PURPOSE:

TO VERIFY THAT INVALID COMMAND STATUS SETS WHEN VOLUME VALID IS INACTIVE DURING A SEARCH COMMAND.

PROCEDURE:

VOLUME VALID IS RESET BY SETTING AND RESETTING DIAGNOSTIC MODE, AFTER WHICH THE TEST EXECUTES A SEARCH COMMAND, VERIFYING THAT "IVC" STATUS SETS.

ABORT SEARCH TEST

PURPOSE:

TO VERIFY THAT THE RM03 INHIBITS A SEARCH WHEN AN ABORT CONDITION EXISTS AT THE START OF THE COMMAND.

PROCEDURE:

THE TEST SETS AN ERROR IN THE ERROR REGISTER AND ISSUES A SEARCH COMMAND, VERIFYING THAT "PIP" REMAINS INACTIVE.

SEARCH AT OFFSET

```
1476                                    PURPOSE:
```

TO VERIFY THAT OFFSET MODE DOES NOT CAUSE SEARCH ERRORS.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND, THEN EXECUTES A SEARCH COMMAND, VERIFYING THAT THERE ARE NO ERRORS DURING THE SEARCH.

```
                                ;PROGRAM REVISION #001
                                
                                .TITLE  DZRMCA - RM03 FUNCTIONAL TEST, PART 1
                                ;*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
```

```
1549            000003                  R3=     %3              ;;GENERAL REGISTER
1550            000004                  R4=     %4              ;;GENERAL REGISTER
1551            000005                  R5=     %5              ;;GENERAL REGISTER
1552            000006                  R6=     %6              ;;GENERAL REGISTER
1553            000007                  R7=     %7              ;;GENERAL REGISTER
1554            000006                  SP=     %6              ;;STACK POINTER
1555            000007                  PC=     %7              ;;PROGRAM COUNTER
1556
1557                                    ;*PRIORITY LEVEL DEFINITIONS
1558            000000                  PR0=    0               ;;PRIORITY LEVEL 0
1559            000040                  PR1=    40              ;;PRIORITY LEVEL 1
1560            000100                  PR2=    100             ;;PRIORITY LEVEL 2
1561            000140                  PR3=    140             ;;PRIORITY LEVEL 3
1562            000200                  PR4=    200             ;;PRIORITY LEVEL 4
1563            000240                  PR5=    240             ;;PRIORITY LEVEL 5
1564            000300                  PR6=    300             ;;PRIORITY LEVEL 6
1565            000340                  PR7=    340             ;;PRIORITY LEVEL 7
1566
1567                                    ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1568            100000                  SW15=   100000
1569            040000                  SW14=   40000
1570            020000                  SW13=   20000
1571            010000                  SW12=   10000
1572            004000                  SW11=   4000
1573            002000                  SW10=   2000
1574            001000                  SW09=   1000
1575            000400                  SW08=   400
1576            000200                  SW07=   200
1577            000100                  SW06=   100
1578            000040                  SW05=   40
1579            000020                  SW04=   20
1580            000010                  SW03=   10
1581            000004                  SW02=   4
1582            000002                  SW01=   2
1583            000001                  SW00=   1
1584                                    .EQUIV  SW09,SW9
1585                                    .EQUIV  SW08,SW8
1586                                    .EQUIV  SW07,SW7
1587                                    .EQUIV  SW06,SW6
1588                                    .EQUIV  SW05,SW5
1589                                    .EQUIV  SW04,SW4
1590                                    .EQUIV  SW03,SW3
1591                                    .EQUIV  SW02,SW2
1592                                    .EQUIV  SW01,SW1
1593                                    .EQUIV  SW00,SW0
1594
1595                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1596            100000                  BIT15=  100000
1597            040000                  BIT14=  40000
1598            020000                  BIT13=  20000
1599            010000                  BIT12=  10000
1600            004000                  BIT11=  4000
1601            002000                  BIT10=  2000
1602            001000                  BIT09=  1000
1603            000400                  BIT08=  400
1604            000200                  BIT07=  200
```

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

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

                                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                                        ;RMCS1  CONTROL STATUS REGISTER

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

                                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

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

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

                                    ;RMDA   DISK ADDRESS REGISTER

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

                                    ;TRACK,SECTOR MASKS

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

                                    ;RMDS   DRIVE STATUS REGISTER

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

```
                                ;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
```

```
1773
1774                                    ;       READ ONLY BITS
1775
1776            100000                  OCC     =       BIT15           ;OCCUPIED
1777            040000                  RG      =       BIT14           ;RUN AND GO
1778            020000                  EBL     =       BIT13           ;END OF BLOCK
1779            010000                  REX     =       BIT12           ;EXCEPTION
1780            004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1781            002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1782            001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1783            000400                  PDA     =       BIT08           ;DATA AREA
1784            000200                  PHA     =       BIT07           ;HEADER AREA
1785            000100                  CONT    =       BIT06           ;CONTINUE
1786            000040                  WC      =       BIT05           ;WORD CLOCK
1787            000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1788            000010                  MWD     =       BIT03           ;WRITE DATA BIT
1789            000004                  LS      =       BIT02           ;LAST SECTOR
1790            000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1791            000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1792
1793                                    ;RMDT   DRIVE TYPE REGISTER
1794
1795            100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1796            040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1797            020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1798            004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1799
1800            020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1801            024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1802
1803                                    ;RMOF   OFFSET REGISTER
1804
1805            010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1806            004000                  ECI     =       BIT11           ;ECC INHIBIT
1807            002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1808            000200                  OFD     =       BIT07           ;OFFSET FORWARD
1809
1810
1811                                    ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1812            001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1813
1814                                    ;RMMR2  MAINTENANCE REGISTER #2
1815
1816                                    ;       READ ONLY BITS
1817            100000                  RQA     =       BIT15           ;PORT A REQUEST
1818            040000                  RQB     =       BIT14           ;PORT B REQUEST
1819            020000                  TAG     =       BIT13           ;TAG CONTROL
1820            010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1821            004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1822            002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1823            001000                  BB09    =       BIT09           ;TAG BUS
1824            000400                  BB08    =       BIT08           ;TAG BUS
1825            000200                  BB07    =       BIT07           ;TAG BUS
1826            000100                  BB06    =       BIT06           ;TAG BUS
1827            000040                  BB05    =       BIT05           ;TAG BUS
1828            000020                  BB04    =       BIT04           ;TAG BUS
```

```
                        BB03    =       BIT03           ;TAG BUS
                        BB02    =       BIT02           ;TAG BUS
                        BB01    =       BIT01           ;TAG BUS
                        BB00    =       BIT00           ;TAG BUS


                        ;RMER2  ERROR REGISTER 2

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

                        .SBTTL  PROGRAM MNEMONICS

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

                        .SBTTL  RM03 REGISTER INDEX VALUES

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


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

```
                        .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER #1

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

                        ;RMCS2  RH CONTROL STATUS REGISTER #2

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

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

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

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

```
  1941           000022                  RMDB    =       22              ;DATA BUFFER
  1942           000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
  1943           000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3
  1944
  1945           176700                  ABASE   =       176700          ;UNIBUS ADDRESS
  1946           120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY
  1947
  1948
  1949                                   .SBTTL  TRAP CATCHER
  1950
  1951           000000                          .=0
  1952                                   ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
  1953                                   ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
  1954                                   ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
  1955           000174                          .=174
  1956  000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
  1957  000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
  1958
  1959
  1960                                   .SBTTL  ACT11 HOOKS
  1961
  1962                                   ;;*****************************************************************
  1963                                   ;HOOKS REQUIRED BY ACT11
  1964           000200                          $SVPC=.                 ;SAVE PC
  1965           000046                          .=46
  1966  000046  042224                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
  1967           000052                          .=52
  1968  000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
  1969           000200                          .=$SVPC                 ;; RESTORE PC
  1970
  1971                                   .SBTTL  STARTING ADDRESS
  1972
  1973                                   ;THE PROGRAM STARTS AT LOCATION 200
  1974           000200                  .       =       200
  1975  000200  000137  005322                  JMP     START           ;JUMP TO START OF PROGRAM
  1976
  1977           001100                  .=1100
  1978                                   .SBTTL  APT PARAMETER BLOCK
  1979
  1980                                   ;;*****************************************************************
  1981                                   ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  1982                                   ;;*****************************************************************
  1983           001100                          .$X=.   ;;SAVE CURRENT LOCATION
  1984           000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
  1985  000024  000200                          200     ;;FOR APT START UP
  1986           000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
  1987  000044  001100                          $APTHDR ;;POINT TO APT HEADER BLOCK
  1988           001100                          .=.$X   ;;RESET LOCATION COUNTER
  1989                                   ;;*****************************************************************
  1990                                   ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
  1991                                   ;INTERFACE SPEC.
  1992
  1993  001100                           $APTHD:
  1994  001100  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
  1995  001102  001222                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
  1996  001104  000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

```
  1997  001106  000002                $PASTM: .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  1998  001110  000002                $UNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  1999  001112  000042                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  2000        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
```

```
2057  001226  000000                $TESTN: .WORD   ATESTN  ;;TEST NUMBER
2058  001230  000000                $PASS:  .WORD   APASS   ;;PASS COUNT
2059  001232  000000                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
2060  001234  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
2061  001236  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
2062  001240  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
2063  001242                        $ETABLE:                ;;APT ENVIRONMENT TABLE
2064  001242     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
2065  001243     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
2066  001244  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
2067  001246  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
2068  001250  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
2069                                ;*                      BITS 15-11=CPU TYPE
2070                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2071                                ;*                              11/70=06,PDQ=07,Q=10
2072                                ;*                      BIT 10=REAL TIME CLOCK
2073                                ;*                      BIT  9=FLOATING POINT PROCESSOR
2074                                ;*                      BIT  8=MEMORY MANAGEMENT
2075  001252     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
2076  001253     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
2077                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
2078                                ;*                              900 NSEC CORE=001
2079                                ;*                              300 NSEC BIPOLAR=002
2080                                ;*                              500 NSEC MOS=003
2081  001254  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
2082                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2083  001256     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
2084  001257     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
2085  001260  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
2086  001262     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
2087  001263     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
2088  001264  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
2089  001266     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
2090  001267     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
2091  001270  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
2092  001272  120254                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
2093  001274  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
2094  001276  176700                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
2095  001300  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
2096  001302  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
2097  001304  000000                $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
2098  001306  000000                $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
2099  001310  000000                $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
2100  001312  000000                $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
2101  001314  000000                $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
2102  001316  000000                $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
2103  001320  000000                $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
2104  001322  000000                $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
2105  001324  000000                $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
2106  001326                        $ETEND:
2107                                .MEXIT
2108
2109                                ;THE REGISTER INPUT BUFFER IS USED FOR
2110                                ;STORING DRIVE STATUS
2111
2112  001326                        GETBUF:
```

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

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

001376                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001376  000000                  RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER
001400  000000                  RMWCO: .WORD                    ;WORD COUNT REGISTER
001402  000000                  RMBAO: .WORD                    ;BUS ADDRESS REGISTER
001404  000000                  RMDAO: .WORD                    ;DISK ADDRESS REGISTER
001406  000000                  RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
001410  000000                  RMDSO: .WORD                    ;DRIVE STATUS REGISTER
001412  000000                  RMER1O: .WORD                   ;ERROR REGISTER 1
001414  000000                  RMASO: .WORD                    ;ATTENTION SUMMARY REGISTER
001416  000000                  RMLAO: .WORD                    ;LOOK AHEAD REGISTER
001420  000000                  RMDBO: .WORD                    ;DATA BUFFER
001422  000000                  RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
001424  000000                  RMDTO: .WORD                    ;DRIVE TYPE REGISTER
001426  000000                  RMSNO: .WORD                    ;SERIAL NUMBER REGISTER
001430  000000                  RMOFO: .WORD                    ;OFFSET REGISTER
001432  000000                  RMDCO: .WORD                    ;DESIRED CYLINDER REGISTER
001434  000000                  RMCCO: .WORD                    ;CURRENT CYLINDER REGISTER
001436  000000                  RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
001440  000000                  RMER2O: .WORD                   ;ERROR REGISTER 2
001442  000000                  RMEC1O: .WORD                   ;ECC POSITION REGISTER
001444  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.

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

```
2169                                    ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
2170                                    ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
2171  001472  000000                    MEDEN8: .WORD           ;MEDIA ENABLE
2172
2173                                    ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
2174                                    ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
2175  001474  000000                    ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
2176  001476  000000                    ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
2177  001500  000000                    CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
2178  001502  000000                    CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
2179
2180                                    ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
2181                                    ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
2182                                    ;A NEGATIVE BYTE.
2183  001504  000027                    GETINX: .BLKB   23.             ;GET INDEX TABLE
2184
2185                                    ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
2186                                    ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
2187                                    ;A NEGATIVE BYTE.
2188  001533  000027                    PUTINX: .BLKB   23.             ;PUT INDEX TABLE
2189
2190                                    ;PUT TAGS HERE
2191
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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

  001562                        $ERRTB:
```

```
2209                                    ;ERROR  1       WRONG UNIT SELECTED
2210  001562  067526                            EMT1
2211  001564  073620                            EHT1
2212  001566  073744                            EDT1
2213  001570  074034                            EFT1
2214
2215
2216                                    ;ERROR  2       DEVICE WENT UNAVAILABLE
2217  001572  067532                            EMT2
2218  001574  073620                            EHT1
2219  001576  073744                            EDT1
2220  001600  074034                            EFT1
2221
2222
2223                                    ;ERROR  3       DEVICE WENT NONEXISTENT
2224  001602  067540                            EMT3
2225  001604  073620                            EHT1
2226  001606  073744                            EDT1
2227  001610  074034                            EFT1
2228
2229
2230                                    ;ERROR  4       CONTROLLER NOT READY
2231  001612  067546                            EMT4
2232  001614  073620                            EHT1
2233  001616  073744                            EDT1
2234  001620  074034                            EFT1
2235
2236
2237                                    ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
2238  001622  067554                            EMT5
2239  001624  073620                            EHT1
2240  001626  073744                            EDT1
2241  001630  074034                            EFT1
2242
2243
2244                                    ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
2245  001632  067562                            EMT6
2246  001634  073620                            EHT1
2247  001636  073744                            EDT1
2248  001640  074034                            EFT1
2249
2250
2251                                    ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
2252  001642  067570                            EMT7
2253  001644  000000                            0
2254  001646  000000                            0
2255  001650  000000                            0
2256
2257
2258                                    ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
2259  001652  067576                            EMT10
2260  001654  073620                            EHT1
2261  001656  073744                            EDT1
2262  001660  074034                            EFT1
2263
2264
```

```
                                ERROR POINTER TABLE

2265
2266  001662  067602            ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
2267  001664  073620                    EMT11
2268  001666  073744                    EHT1
2269  001670  074034                    EDT1
2270                                     EFT1
2271
2272                            ;ERROR  12      INCORRECT FUNCTION CODE
2273  001672  067606                    EMT12
2274  001674  073620                    EHT1
2275  001676  073744                    EDT1
2276  001700  074034                    EFT1
2277
2278
2279                            ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
2280  001702  067614                    EMT13
2281  001704  073620                    EHT1
2282  001706  073744                    EDT1
2283  001710  074034                    EFT1
2284
2285
2286                            ;ERROR  14      TRANSFER ERROR IS INCORRECT
2287  001712  067626                    EMT14
2288  001714  073620                    EHT1
2289  001716  073744                    EDT1
2290  001720  074034                    EFT1
2291
2292
2293                            ;ERROR  15      INCORRECT WORD COUNT
2294  001722  067634                    EMT15
2295  001724  073620                    EHT1
2296  001726  073744                    EDT1
2297  001730  074034                    EFT1
2298
2299
2300                            ;ERROR  16      INCORRECT BUS ADDRESS
2301  001732  067642                    EMT16
2302  001734  073620                    EHT1
2303  001736  073744                    EDT1
2304  001740  074034                    EFT1
2305
2306
2307                            ;ERROR  17      INCORRECT LBT STATUS
2308  001742  067652                    EMT17
2309  001744  073620                    EHT1
2310  001746  073744                    EDT1
2311  001750  074034                    EFT1
2312
2313
2314                            ;ERROR  20      INCORRECT AOE
2315  001752  067662                    EMT20
2316  001754  073620                    EHT1
2317  001756  073744                    EDT1
2318  001760  074034                    EFT1
2319
2320
```

ERROR POINTER TABLE

```
2321
2322  001762  067672          ;ERROR  21      INCORRECT DISK ADDRESS
2323  001764  073620                  EMT21
2324  001766  073744                  EHT1
2325  001770  074034                  EDT1
2326                            EFT1
2327
2328                          ;ERROR  22      INCORRECT CYLINDER ADDRESS
2329  001772  067702                  EMT22
2330  001774  073620                  EHT1
2331  001776  073744                  EDT1
2332  002000  074034                  EFT1
2333
2334
2335                          ;ERROR  23      INCORRECT WLE STATUS
2336  002002  067712                  EMT23
2337  002004  073620                  EHT1
2338  002006  073744                  EDT1
2339  002010  074034                  EFT1
2340
2341
2342                          ;ERROR  24      INCORRECT UPE STATUS
2343  002012  067722                  EMT24
2344  002014  073620                  EHT1
2345  002016  073744                  EDT1
2346  002020  074034                  EFT1
2347
2348
2349                          ;ERROR  25      INCORRECT WCF STATUS
2350  002022  067732                  EMT25
2351  002024  073620                  EHT1
2352  002026  073744                  EDT1
2353  002030  074034                  EFT1
2354
2355
2356                          ;ERROR  26      INCORRECT WCE STATUS
2357  002032  067742                  EMT26
2358  002034  073620                  EHT1
2359  002036  073744                  EDT1
2360  002040  074034                  EFT1
2361
2362
2363                          ;ERROR  27      INCORRECT MDPE STATUS
2364  002042  067752                  EMT27
2365  002044  073620                  EHT1
2366  002046  073744                  EDT1
2367  002050  074034                  EFT1
2368
2369
2370                          ;ERROR  30      INCORRECT DCK STATUS
2371  002052  067762                  EMT30
2372  002054  073620                  EHT1
2373  002056  073744                  EDT1
2374  002060  074034                  EFT1
2375
2376
```

```
2377
2378  002062  067772                ;ERROR  31      INCORRECT ECH STATUS
2379  002064  073620                        EMT31
2380  002066  073744                        EHT1
2381  002070  074034                        EDT1
2382                                  EFT1
2383
2384                                ;ERROR  32      DLT SHOULD NOT BE SET
2385  002072  070002                        EMT32
2386  002074  073620                        EHT1
2387  002076  073744                        EDT1
2388  002100  074034                        EFT1
2389
2390
2391                                ;ERROR  33      MXF SHOULD NOT BE SET
2392  002102  070012                        EMT33
2393  002104  073620                        EHT1
2394  002106  073744                        EDT1
2395  002110  074034                        EFT1
2396
2397
2398                                ;ERROR  34      DTE SHOULD NOT BE SET
2399  002112  070022                        EMT34
2400  002114  073620                        EHT1
2401  002116  073744                        EDT1
2402  002120  074034                        EFT1
2403
2404
2405                                ;ERROR  35      INCORRECT HCRC STATUS
2406  002122  070032                        EMT35
2407  002124  073620                        EHT1
2408  002126  073744                        EDT1
2409  002130  074034                        EFT1
2410
2411
2412                                ;ERROR  36      INCORRECT HCE STATUS
2413  002132  070042                        EMT36
2414  002134  073620                        EHT1
2415  002136  073744                        EDT1
2416  002140  074034                        EFT1
2417
2418
2419                                ;ERROR  37      INCORRECT FER STATUS
2420  002142  070052                        EMT37
2421  002144  073620                        EHT1
2422  002146  073744                        EDT1
2423  002150  074034                        EFT1
2424
2425
2426                                ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
2427  002152  070062                        EMT40
2428  002154  073620                        EHT1
2429  002156  073744                        EDT1
2430  002160  074034                        EFT1
2431
2432
```

```
2433
2434  002162  070070                ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
2435  002164  073620                        EMT41
2436  002166  073744                        EHT1
2437  002170  074034                        EDT1
2438                                  EFT1
2439
2440                                ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
2441  002172  070100                        EMT42
2442  002174  073620                        EHT1
2443  002176  073744                        EDT1
2444  002200  074034                        EFT1
2445
2446
2447                                ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
2448  002202  070112                        EMT43
2449  002204  073620                        EHT1
2450  002206  073744                        EDT1
2451  002210  074034                        EFT1
2452
2453
2454                                ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
2455  002212  070122                        EMT44
2456  002214  073620                        EHT1
2457  002216  073744                        EDT1
2458  002220  074034                        EFT1
2459
2460
2461                                ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
2462  002222  070132                        EMT45
2463  002224  073620                        EHT1
2464  002226  073744                        EDT1
2465  002230  074034                        EFT1
2466
2467
2468                                ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
2469  002232  070142                        EMT46
2470  002234  073620                        EHT1
2471  002236  073744                        EDT1
2472  002240  074034                        EFT1
2473
2474
2475                                ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
2476  002242  070152                        EMT47
2477  002244  073620                        EHT1
2478  002246  073744                        EDT1
2479  002250  074034                        EFT1
2480
2481
2482                                ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
2483  002252  070160                        EMT50
2484  002254  073620                        EHT1
2485  002256  073744                        EDT1
2486  002260  074034                        EFT1
2487
2488
```

```
2489
2490  002262  070170                ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
2491  002264  073620                        EMT51
2492  002266  073744                        EHT1
2493  002270  074034                        EDT1
2494                                  EFT1
2495
2496                                ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
2497  002272  070202                        EMT52
2498  002274  073620                        EHT1
2499  002276  073744                        EDT1
2500  002300  074034                        EFT1
2501
2502
2503                                ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
2504                                ;               ON CYLINDER LATCH DIDN'T RESET
2505  002302  070220                        EMT53
2506  002304  073620                        EHT1
2507  002306  073744                        EDT1
2508  002310  074034                        EFT1
2509
2510
2511                                ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
2512  002312  070236                        EMT54
2513  002314  073620                        EHT1
2514  002316  073744                        EDT1
2515  002320  074034                        EFT1
2516
2517
2518                                ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
2519  002322  070246                        EMT55
2520  002324  073620                        EHT1
2521  002326  073744                        EDT1
2522  002330  074034                        EFT1
2523
2524
2525                                ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
2526  002332  070260                        EMT56
2527  002334  073620                        EHT1
2528  002336  073744                        EDT1
2529  002340  074034                        EFT1
2530
2531
2532                                ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
2533  002342  070276                        EMT57
2534  002344  073620                        EHT1
2535  002346  073744                        EDT1
2536  002350  074034                        EFT1
2537
2538
2539                                ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
2540                                ;               VOLUME VALID
2541  002352  070306                        EMT60
2542  002354  073620                        EHT1
2543  002356  073744                        EDT1
2544  002360  074034                        EFT1
```

ERROR POINTER TABLE

```
2545
2546
2547                                    ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
2548                                    ;               VOLUME VALID IS STIL SET
2549  002362  070324                                    EMT61
2550  002364  073620                                    EHT1
2551  002366  073744                                    EDT1
2552  002370  074034                                    EFT1
2553
2554
2555                                    ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
2556                                    ;               REPORTED DURING SEEK COMMAND
2557  002372  070344                                    EMT62
2558  002374  073620                                    EHT1
2559  002376  073744                                    EDT1
2560  002400  074034                                    EFT1
2561
2562
2563                                    ;ERROR  63      UNUSED
2564  002402  000000                                    0
2565  002404  000000                                    0
2566  002406  000000                                    0
2567  002410  000000                                    0
2568
2569
2570                                    ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
2571  002412  070362                                    EMT64
2572  002414  073620                                    EHT1
2573  002416  073744                                    EDT1
2574  002420  074034                                    EFT1
2575
2576
2577                                    ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
2578  002422  070402                                    EMT65
2579  002424  073620                                    EHT1
2580  002426  073744                                    EDT1
2581  002430  074034                                    EFT1
2582
2583
2584                                    ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
2585  002432  070422                                    EMT66
2586  002434  073620                                    EHT1
2587  002436  073744                                    EDT1
2588  002440  074034                                    EFT1
2589
2590
2591                                    ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
2592  002442  070434                                    EMT67
2593  002444  073620                                    EHT1
2594  002446  073744                                    EDT1
2595  002450  074034                                    EFT1
2596
2597
2598                                    ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
2599  002452  070446                                    EMT70
2600  002454  073620                                    EHT1
```

```
 2601  002456  073744                    EDT1
 2602  002460  074034                    EFT1
 2603
 2604
 2605                              ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
 2606  002462  070456                    EMT71
 2607  002464  073620                    EHT1
 2608  002466  073744                    EDT1
 2609  002470  074034                    EFT1
 2610
 2611
 2612                              ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
 2613  002472  070466                    EMT72
 2614  002474  073620                    EHT1
 2615  002476  073744                    EDT1
 2616  002500  074034                    EFT1
 2617
 2618
 2619                              ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
 2620                              ;               CYLINDER DIDNT DROP
 2621  002502  070504                    EMT73
 2622  002504  073620                    EHT1
 2623  002506  073744                    EDT1
 2624  002510  074034                    EFT1
 2625
 2626
 2627                              ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
 2628  002512  070522                    EMT74
 2629  002514  073620                    EHT1
 2630  002516  073744                    EDT1
 2631  002520  074034                    EFT1
 2632
 2633
 2634                              ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
 2635  002522  070532                    EMT75
 2636  002524  073620                    EHT1
 2637  002526  073744                    EDT1
 2638  002530  074034                    EFT1
 2639
 2640
 2641                              ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
 2642  002532  070552                    EMT76
 2643  002534  073620                    EHT1
 2644  002536  073744                    EDT1
 2645  002540  074034                    EFT1
 2646
 2647
 2648                              ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
 2649  002542  070562                    EMT77
 2650  002544  073620                    EHT1
 2651  002546  073744                    EDT1
 2652  002550  074034                    EFT1
 2653
 2654
 2655                              ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
 2656  002552  070574                    EMT100
```

```
 2657  002554  073620                    EHT1
 2658  002556  073744                    EDT1
 2659  002560  074034                    EFT1
 2660
 2661
 2662                              ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
 2663  002562  070612                    EMT101
 2664  002564  073620                    EHT1
 2665  002566  073744                    EDT1
 2666  002570  074034                    EFT1
 2667
 2668
 2669                              ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
 2670  002572  070630                    EMT102
 2671  002574  073620                    EHT1
 2672  002576  073744                    EDT1
 2673  002600  074034                    EFT1
 2674
 2675
 2676                              ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
 2677  002602  070640                    EMT103
 2678  002604  073620                    EHT1
 2679  002606  073744                    EDT1
 2680  002610  074034                    EFT1
 2681
 2682
 2683                              ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
 2684  002612  070652                    EMT104
 2685  002614  073620                    EHT1
 2686  002616  073744                    EDT1
 2687  002620  074034                    EFT1
 2688
 2689
 2690                              ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
 2691  002622  070670                    EMT105
 2692  002624  073620                    EHT1
 2693  002626  073744                    EDT1
 2694  002630  074034                    EFT1
 2695
 2696
 2697                              ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
 2698  002632  070700                    EMT106
 2699  002634  073620                    EHT1
 2700  002636  073744                    EDT1
 2701  002640  074034                    EFT1
 2702
 2703
 2704                              ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
 2705  002642  070710                    EMT107
 2706  002644  073620                    EHT1
 2707  002646  073744                    EDT1
 2708  002650  074034                    EFT1
 2709
 2710
 2711                              ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
 2712  002652  070730                    EMT110
```

ERROR POINTER TABLE

```
2713  002654  073642                    EHT110
2714  002656  073762                    EDT110
2715  002660  074052                    EFT110
2716
2717
2718                          ;ERROR  111     NONEXISTENT DEVICE
2719  002662  070742                    EMT111
2720  002664  073646                    EHT111
2721  002666  073764                    EDT111
2722  002670  074054                    EFT111
2723
2724
2725                          ;ERROR  112     DEVICE NOT AVAILABLE
2726  002672  070750                    EMT112
2727  002674  073646                    EHT111
2728  002676  073764                    EDT111
2729  002700  074054                    EFT111
2730
2731
2732                          ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
2733  002702  070756                    EMT113
2734  002704  000000                    0
2735  002706  000000                    0
2736  002710  000000                    0
2737
2738
2739                          ;ERROR  114     DEVICE NOT AN RM03
2740  002712  070772                    EMT114
2741  002714  073652                    EHT114
2742  002716  073766                    EDT114
2743  002720  074056                    EFT114
2744
2745
2746                          ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
2747  002722  071000                    EMT115
2748  002724  073620                    EHT1
2749  002726  073744                    EDT1
2750  002730  074034                    EFT1
2751
2752
2753                          ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
2754  002732  071010                    EMT116
2755  002734  073620                    EHT1
2756  002736  073744                    EDT1
2757  002740  074034                    EFT1
2758
2759
2760                          ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
2761  002742  071020                    EMT117
2762  002744  073620                    EHT1
2763  002746  073744                    EDT1
2764  002750  074034                    EFT1
2765
2766
2767                          ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
2768  002752  071030                    EMT120
```

```
 2769  002754  073620                    EHT1
 2770  002756  073744                    EDT1
 2771  002760  074034                    EFT1
 2772
 2773
 2774                          ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
 2775  002762  071040                    EMT121
 2776  002764  073620                    EHT1
 2777  002766  073744                    EDT1
 2778  002770  074034                    EFT1
 2779
 2780
 2781                          ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
 2782  002772  071050                    EMT122
 2783  002774  073620                    EHT1
 2784  002776  073744                    EDT1
 2785  003000  074034                    EFT1
 2786
 2787
 2788                          ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
 2789  003002  071060                    EMT123
 2790  003004  073620                    EHT1
 2791  003006  073744                    EDT1
 2792  003010  074034                    EFT1
 2793
 2794
 2795                          ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
 2796  003012  071070                    EMT124
 2797  003014  073620                    EHT1
 2798  003016  073744                    EDT1
 2799  003020  074034                    EFT1
 2800
 2801
 2802                          ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
 2803  003022  071100                    EMT125
 2804  003024  073620                    EHT1
 2805  003026  073744                    EDT1
 2806  003030  074034                    EFT1
 2807
 2808
 2809                          ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
 2810  003032  071110                    EMT126
 2811  003034  073620                    EHT1
 2812  003036  073744                    EDT1
 2813  003040  074034                    EFT1
 2814
 2815
 2816                          ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
 2817  003042  071122                    EMT127
 2818  003044  073620                    EHT1
 2819  003046  073744                    EDT1
 2820  003050  074034                    EFT1
 2821
 2822
 2823                          ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
 2824  003052  071134                    EMT130
```

ERROR POINTER TABLE

```
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
                                        EMT131
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
                                        EMT132
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
                                        EMT133
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
                                        EMT134
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
                                        EMT135
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
                                        EMT136
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
                                        EMT137
                                        EHT1
                                        EDT1
                                        EFT1

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

ERROR POINTER TABLE

```
  2881  003154  073620                    EHT1
  2882  003156  073744                    EDT1
  2883  003160  074034                    EFT1
  2884
  2885
  2886                          ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
  2887  003162  071262                    EMT141
  2888  003164  073620                    EHT1
  2889  003166  073744                    EDT1
  2890  003170  074034                    EFT1
  2891
  2892
  2893                          ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
  2894  003172  071272                    EMT142
  2895  003174  073620                    EHT1
  2896  003176  073744                    EDT1
  2897  003200  074034                    EFT1
  2898
  2899
  2900                          ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
  2901  003202  071302                    EMT143
  2902  003204  073620                    EHT1
  2903  003206  073744                    EDT1
  2904  003210  074034                    EFT1
  2905
  2906
  2907                          ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
  2908  003212  071312                    EMT144
  2909  003214  073620                    EHT1
  2910  003216  073744                    EDT1
  2911  003220  074034                    EFT1
  2912
  2913
  2914                          ;ERROR  145     RMMR1  NOT CLEARED BY DPIVE CLEAR
  2915  003222  071322                    EMT145
  2916  003224  073620                    EHT1
  2917  003226  073744                    EDT1
  2918  003230  074034                    EFT1
  2919
  2920
  2921                          ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
  2922  003232  071332                    EMT146
  2923  003234  073620                    EHT1
  2924  003236  073744                    EDT1
  2925  003240  074034                    EFT1
  2926
  2927
  2928                          ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
  2929  003242  071342                    EMT147
  2930  003244  073620                    EHT1
  2931  003246  073744                    EDT1
  2932  003250  074034                    EFT1
  2933
  2934
  2935                          ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
  2936  003252  071352                    EMT150
```

```
  2937  003254  073620                    EHT1
  2938  003256  073744                    EDT1
  2939  003260  074034                    EFT1
  2940
  2941
  2942                              ;ERROR  151     MEDIUM NOT ON LINE
  2943  003262  071362                    EMT151
  2944  003264  073620                    EHT1
  2945  003266  073744                    EDT1
  2946  003270  074034                    EFT1
  2947
  2948
  2949                              ;ERROR  152     DRIVE FAULT
  2950  003272  071374                    EMT152
  2951  003274  073620                    EHT1
  2952  003276  073744                    EDT1
  2953  003300  074034                    EFT1
  2954
  2955
  2956                              ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
  2957  003302  071406                    EMT153
  2958  003304  073620                    EHT1
  2959  003306  073744                    EDT1
  2960  003310  074034                    EFT1
  2961
  2962
  2963                              ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
  2964  003312  071424                    EMT154
  2965  003314  073620                    EHT1
  2966  003316  073744                    EDT1
  2967  003320  074034                    EFT1
  2968
  2969
  2970                              ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
  2971  003322  071442                    EMT155
  2972  003324  073620                    EHT1
  2973  003326  073744                    EDT1
  2974  003330  074034                    EFT1
  2975
  2976
  2977                              ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
  2978  003332  071454                    EMT156
  2979  003334  073620                    EHT1
  2980  003336  073744                    EDT1
  2981  003340  074034                    EFT1
  2982
  2983
  2984                              ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
  2985  003342  071466                    EMT157
  2986  003344  073620                    EHT1
  2987  003346  073744                    EDT1
  2988  003350  074034                    EFT1
  2989
  2990
  2991                              ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
  2992  003352  071500                    EMT160
```

```
 2993  003354  073620                    EHT1
 2994  003356  073744                    EDT1
 2995  003360  074034                    EFT1
 2996
 2997
 2998                          ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
 2999  003362  071510                    EMT161
 3000  003364  073620                    EHT1
 3001  003366  073744                    EDT1
 3002  003370  074034                    EFT1
 3003
 3004
 3005                          ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
 3006  003372  071522                    EMT162
 3007  003374  073620                    EHT1
 3008  003376  073744                    EDT1
 3009  003400  074034                    EFT1
 3010
 3011
 3012                          ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
 3013                          ;               WRITE BUFFER
 3014  003402  073414                    EMT336
 3015  003404  073702                    EHT336
 3016  003406  074000                    EDT336
 3017  003410  074070                    EFT336
 3018
 3019
 3020                          ;ERROR  164     OPI SHOULD NOT BE SET
 3021  003412  071544                    EMT164
 3022  003414  073620                    EHT1
 3023  003416  073744                    EDT1
 3024  003420  074034                    EFT1
 3025
 3026
 3027                          ;ERROR  165     IVC SHOULD NOT BE SET
 3028  003422  071552                    EMT165
 3029  003424  073620                    EHT1
 3030  003426  073744                    EDT1
 3031  003430  074034                    EFT1
 3032
 3033
 3034                          ;ERROR  166     IAE SHOULD NOT BE SET
 3035  003432  071560                    EMT166
 3036  003434  073620                    EHT1
 3037  003436  073744                    EDT1
 3038  003440  074034                    EFT1
 3039
 3040
 3041                          ;ERROR  167     NEM SHOULD NOT BE SET
 3042  003442  071566                    EMT167
 3043  003444  073620                    EHT1
 3044  003446  073744                    EDT1
 3045  003450  074034                    EFT1
 3046
 3047
 3048                          ;ERROR 170      UNUSED
```

```
 3049  003452  000000                          0
 3050  003454  000000                          0
 3051  003456  000000                          0
 3052  003460  000000                          0
 3053
 3054
 3055                          ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
 3056  003462  071604                          EMT171
 3057  003464  073620                          EHT1
 3058  003466  073744                          EDT1
 3059  003470  074034                          EFT1
 3060
 3061
 3062                          ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
 3063  003472  071614                          EMT172
 3064  003474  073620                          EHT1
 3065  003476  073744                          EDT1
 3066  003500  074034                          EFT1
 3067
 3068
 3069                          ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
 3070  003502  071624                          EMT173
 3071  003504  073620                          EHT1
 3072  003506  073744                          EDT1
 3073  003510  074034                          EFT1
 3074
 3075
 3076                          ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
 3077  003512  071634                          EMT174
 3078  003514  073620                          EHT1
 3079  003516  073744                          EDT1
 3080  003520  074034                          EFT1
 3081
 3082
 3083                          ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
 3084  003522  071646                          EMT175
 3085  003524  073620                          EHT1
 3086  003526  073744                          EDT1
 3087  003530  074034                          EFT1
 3088
 3089
 3090                          ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
 3091  003532  071654                          EMT176
 3092  003534  073620                          EHT1
 3093  003536  073744                          EDT1
 3094  003540  074034                          EFT1
 3095
 3096
 3097                          ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
 3098  003542  071662                          EMT177
 3099  003544  073620                          EHT1
 3100  003546  073744                          EDT1
 3101  003550  074034                          EFT1
 3102
 3103
 3104                          ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
 3105  003552  071674                    EMT200
 3106  003554  073620                    EHT1
 3107  003556  073744                    EDT1
 3108  003560  074034                    EFT1
 3109
 3110
 3111                          ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
 3112  003562  071706                    EMT201
 3113  003564  073620                    EHT1
 3114  003566  073744                    EDT1
 3115  003570  074034                    EFT1
 3116
 3117
 3118                          ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
 3119  003572  071720                    EMT202
 3120  003574  073620                    EHT1
 3121  003576  073744                    EDT1
 3122  003600  074034                    EFT1
 3123
 3124
 3125                          ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
 3126  003602  071732                    EMT203
 3127  003604  073620                    EHT1
 3128  003606  073744                    EDT1
 3129  003610  074034                    EFT1
 3130
 3131
 3132                          ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
 3133  003612  071744                    EMT204
 3134  003614  073620                    EHT1
 3135  003616  073744                    EDT1
 3136  003620  074034                    EFT1
 3137
 3138
 3139                          ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
 3140  003622  071762                    EMT205
 3141  003624  073620                    EHT1
 3142  003626  073744                    EDT1
 3143  003630  074034                    EFT1
 3144
 3145
 3146                          ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
 3147  003632  071772                    EMT206
 3148  003634  073620                    EHT1
 3149  003636  073744                    EDT1
 3150  003640  074034                    EFT1
 3151
 3152
 3153                          ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
 3154  003642  072002                    EMT207
 3155  003644  073620                    EHT1
 3156  003646  073744                    EDT1
 3157  003650  074034                    EFT1
 3158
 3159
 3160                          ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
3161  003652  072014                        EMT210
3162  003654  073620                        EHT1
3163  003656  073744                        EDT1
3164  003660  074034                        EFT1
3165
3166
3167                                ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
3168  003662  072022                        EMT211
3169  003664  073620                        EHT1
3170  003666  073744                        EDT1
3171  003670  074034                        EFT1
3172
3173
3174                                ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
3175  003672  072036                        EMT212
3176  003674  073620                        EHT1
3177  003676  073744                        EDT1
3178  003700  074034                        EFT1
3179
3180
3181                                ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
3182  003702  072052                        EMT213
3183  003704  073620                        EHT1
3184  003706  073744                        EDT1
3185  003710  074034                        EFT1
3186
3187
3188                                ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
3189  003712  072062                        EMT214
3190  003714  073632                        EHT2
3191  003716  073754                        EDT2
3192  003720  074044                        EFT2
3193
3194
3195                                ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
3196  003722  072102                        EMT215
3197  003724  073632                        EHT2
3198  003726  073754                        EDT2
3199  003730  074044                        EFT2
3200
3201
3202                                ;ERROR  216     INCORRECT "IVC" STATUS
3203  003732  072114                        EMT216
3204  003734  073620                        EHT1
3205  003736  073744                        EDT1
3206  003740  074034                        EFT1
3207
3208
3209                                ;ERROR  217     INCORRECT "IAE" STATUS
3210  003742  072124                        EMT217
3211  003744  073620                        EHT1
3212  003746  073744                        EDT1
3213  003750  074034                        EFT1
3214
3215
3216                                ;ERROR 220      INCORRECT "WLE" STATUS
```

ERROR POINTER TABLE

```
3217  003752  072134                    EMT220
3218  003754  073620                    EHT1
3219  003756  073744                    EDT1
3220  003760  074034                    EFT1
3221
3222
3223                          ;ERROR  221     INCORRECT "OPI" STATUS
3224  003762  072144                    EMT221
3225  003764  073620                    EHT1
3226  003766  073744                    EDT1
3227  003770  074034                    EFT1
3228
3229
3230                          ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
3231  003772  072154                    EMT222
3232  003774  073620                    EHT1
3233  003776  073744                    EDT1
3234  004000  074034                    EFT1
3235
3236
3237                          ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
3238  004002  072164                    EMT223
3239  004004  073656                    EHT223
3240  004006  073770                    EDT223
3241  004010  074060                    EFT223
3242
3243
3244                          ;ERROR  224     UNUSED
3245  004012  000000                    0
3246  004014  000000                    0
3247  004016  000000                    0
3248  004020  000000                    0
3249
3250
3251                          ;ERROR  225     UNUSED
3252  004022  000000                    0
3253  004024  000000                    0
3254  004026  000000                    0
3255  004030  000000                    0
3256
3257
3258                          ;ERROR  226     UNUSED
3259  004032  000000                    0
3260  004034  000000                    0
3261  004036  000000                    0
3262  004040  000000                    0
3263
3264
3265                          ;ERROR  227     UNUSED
3266  004042  000000                    0
3267  004044  000000                    0
3268  004046  000000                    0
3269  004050  000000                    0
3270
3271
3272                          ;ERROR  230     UNUSED
```

```
3273  004052  000000                    0
3274  004054  000000                    0
3275  004056  000000                    0
3276  004060  000000                    0
3277
3278
3279                              ;ERROR  231     UNUSED
3280  004062  000000                    0
3281  004064  000000                    0
3282  004066  000000                    0
3283  004070  000000                    0
3284
3285
3286                              ;ERROR  232     UNUSED
3287  004072  000000                    0
3288  004074  000000                    0
3289  004076  000000                    0
3290  004100  000000                    0
3291
3292
3293                              ;ERROR  233     UNUSED
3294  004102  000000                    0
3295  004104  000000                    0
3296  004106  000000                    0
3297  004110  000000                    0
3298
3299
3300                              ;ERROR  234     UNUSED
3301  004112  000000                    0
3302  004114  000000                    0
3303  004116  000000                    0
3304  004120  000000                    0
3305
3306
3307                              ;ERROR  235     UNUSED
3308  004122  000000                    0
3309  004124  000000                    0
3310  004126  000000                    0
3311  004130  000000                    0
3312
3313
3314                              ;ERROR  236     UNUSED
3315  004132  000000                    0
3316  004134  000000                    0
3317  004136  000000                    0
3318  004140  000000                    0
3319
3320
3321                              ;ERROR  237     UNUSED
3322  004142  000000                    0
3323  004144  000000                    0
3324  004146  000000                    0
3325  004150  000000                    0
3326
3327
3328                              ;ERROR  240     UNUSED
```

```
3329  004152  000000                    0
3330  004154  000000                    0
3331  004156  000000                    0
3332  004160  000000                    0
3333
3334
3335                          ;ERROR  241     UNUSED
3336  004162  000000                    0
3337  004164  000000                    0
3338  004166  000000                    0
3339  004170  000000                    0
3340
3341
3342                          ;ERROR  242     UNUSED
3343  004172  000000                    0
3344  004174  000000                    0
3345  004176  000000                    0
3346  004200  000000                    0
3347
3348
3349                          ;ERROR  243     UNUSED
3350  004202  000000                    0
3351  004204  000000                    0
3352  004206  000000                    0
3353  004210  000000                    0
3354
3355
3356                          ;ERROR  244     UNUSED
3357  004212  000000                    0
3358  004214  000000                    0
3359  004216  000000                    0
3360  004220  000000                    0
3361
3362
3363                          ;ERROR  245     UNUSED
3364  004222  000000                    0
3365  004224  000000                    0
3366  004226  000000                    0
3367  004230  000000                    0
3368
3369
3370                          ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
3371  004232  072240                    EMT246
3372  004234  073620                    EHT1
3373  004236  073744                    EDT1
3374  004240  074034                    EFT1
3375
3376
3377                          ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
3378  004242  072250                    EMT247
3379  004244  073620                    EHT1
3380  004246  073744                    EDT1
3381  004250  074034                    EFT1
3382
3383
3384                          ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
  3385  004252  072262                    EMT250
  3386  004254  073620                    EHT1
  3387  004256  073744                    EDT1
  3388  004260  074034                    EFT1
  3389
  3390
  3391                          ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
  3392  004262  072276                    EMT251
  3393  004264  073632                    EHT2
  3394  004266  073754                    EDT2
  3395  004270  074044                    EFT2
  3396
  3397
  3398                          ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
  3399  004272  072304                    EMT252
  3400  004274  073632                    EHT2
  3401  004276  073754                    EDT2
  3402  004300  074044                    EFT2
  3403
  3404
  3405                          ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
  3406  004302  072312                    EMT253
  3407  004304  073620                    EHT1
  3408  004306  073744                    EDT1
  3409  004310  074034                    EFT1
  3410
  3411
  3412                          ;ERROR  254     INCORRECT "ILF" STATUS
  3413  004312  072330                    EMT254
  3414  004314  073620                    EHT1
  3415  004316  073744                    EDT1
  3416  004320  074034                    EFT1
  3417
  3418
  3419                          ;ERROR  255     INCORRECT "ATA" STATUS
  3420  004322  072340                    EMT255
  3421  004324  073620                    EHT1
  3422  004326  073744                    EDT1
  3423  004330  074034                    EFT1
  3424
  3425
  3426                          ;ERROR  256     INCORRECT "ILR" STATUS
  3427  004332  072350                    EMT256
  3428  004334  073670                    EHT256
  3429  004336  073770                    EDT223
  3430  004340  074060                    EFT223
  3431
  3432
  3433                          ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
  3434  004342  072360                    EMT257
  3435  004344  073620                    EHT1
  3436  004346  073744                    EDT1
  3437  004350  074034                    EFT1
  3438
  3439
  3440                          ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
3441  004352  072372                        EMT260
3442  004354  073620                        EHT1
3443  004356  073744                        EDT1
3444  004360  074034                        EFT1
3445
3446
3447                                ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
3448  004362  072404                        EMT261
3449  004364  073620                        EHT1
3450  004366  073744                        EDT1
3451  004370  074034                        EFT1
3452
3453
3454                                ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
3455  004372  072424                        EMT262
3456  004374  073620                        EHT1
3457  004376  073744                        EDT1
3458  004400  074034                        EFT1
3459
3460
3461                                ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
3462  004402  072434                        EMT263
3463  004404  073620                        EHT1
3464  004406  073744                        EDT1
3465  004410  074034                        EFT1
3466
3467
3468                                ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
3469  004412  072444                        EMT264
3470  004414  073620                        EHT1
3471  004416  073744                        EDT1
3472  004420  074034                        EFT1
3473
3474
3475                                ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
3476  004422  072464                        EMT265
3477  004424  073520                        EHT1
3478  004426  073744                        EDT1
3479  004430  074034                        EFT1
3480
3481
3482                                ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
3483  004432  072504                        EMT266
3484  004434  073620                        EHT1
3485  004436  073744                        EDT1
3486  004440  074034                        EFT1
3487
3488
3489                                ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
3490                                ;               ADDRESS IS TOO LARGE
3491  004442  072516                        EMT267
3492  004444  073620                        EHT1
3493  004446  073744                        EDT1
3494  004450  074034                        EFT1
3495
3496
```

```
3497
3498  004452  072534                ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
3499  004454  073620                        EMT270
3500  004456  073744                        EHT1
3501  004460  074034                        EDT1
3502                                  EFT1
3503
3504                                ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
3505                                ;               DIDN'T DROP
3506  004462  072550                        EMT271
3507  004464  073620                        EHT1
3508  004466  073744                        EDT1
3509  004470  074034                        EFT1
3510
3511
3512                                ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
3513  004472  072566                        EMT272
3514  004474  073620                        EHT1
3515  004476  073744                        EDT1
3516  004500  074034                        EFT1
3517
3518
3519                                ;ERROR  273     PIP STIL SET AFTER SEARCH
3520  004502  072604                        EMT273
3521  004504  073620                        EHT1
3522  004506  073744                        EDT1
3523  004510  074034                        EFT1
3524
3525
3526                                ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
3527                                ;               REGISTERS BUT MXF DID NOT SET
3528  004512  072622                        EMT274
3529  004514  073620                        EHT1
3530  004516  073744                        EDT1
3531  004520  074034                        EFT1
3532
3533
3534                                ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
3535                                ;               COMMAND STARTED
3536  004522  072640                        EMT275
3537  004524  073620                        EHT1
3538  004526  073744                        EDT1
3539  004530  074034                        EFT1
3540
3541
3542                                ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
3543                                ;               ZERO
3544  004532  072652                        EMT276
3545  004534  073620                        EHT1
3546  004536  073744                        EDT1
3547  004540  074034                        EFT1
3548
3549
3550                                ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
3551                                ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
3552                                ;               3) RUN TIMED OUT
```

```
3553  004542  072666                  EMT277
3554  004544  073620                  EHT1
3555  004546  073744                  EDT1
3556  004550  074034                  EFT1
3557
3558
3559                          ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
3560                          ;               WAS NOT VALID
3561  004552  072704                  EMT300
3562  004554  073620                  EHT1
3563  004556  073744                  EDT1
3564  004560  074034                  EFT1
3565
3566
3567                          ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
3568                          ;               IS VALID
3569  004562  072724                  EMT301
3570  004564  073620                  EHT1
3571  004566  073744                  EDT1
3572  004570  074034                  EFT1
3573
3574
3575                          ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
3576  004572  072746                  EMT302
3577  004574  073620                  EHT1
3578  004576  073744                  EDT1
3579  004600  074034                  EFT1
3580
3581
3582                          ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
3583  004602  072760                  EMT303
3584  004604  073620                  EHT1
3585  004606  073744                  EDT1
3586  004610  074034                  EFT1
3587
3588
3589                          ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
3590  004612  072772                  EMT304
3591  004614  073620                  EHT1
3592  004616  073744                  EDT1
3593  004620  074034                  EFT1
3594
3595
3596                          ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
3597  004622  073004                  EMT305
3598  004624  073620                  EHT1
3599  004626  073744                  EDT1
3600  004630  074034                  EFT1
3601
3602
3603                          ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
3604  004632  073016                  EMT306
3605  004634  073620                  EHT1
3606  004636  073744                  EDT1
3607  004640  074034                  EFT1
3608
```

```
3609
3610                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
3611  004642  073026                              EMT307
3612  004644  073620                              EHT1
3613  004646  073744                              EDT1
3614  004650  074034                              EFT1
3615
3616
3617                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
3618  004652  073046                              EMT310
3619  004654  073620                              EHT1
3620  004656  073744                              EDT1
3621  004660  074034                              EFT1
3622
3623
3624                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
3625  004662  073060                              EMT311
3626  004664  073620                              EHT1
3627  004666  073744                              EDT1
3628  004670  074034                              EFT1
3629
3630
3631                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
3632  004672  073072                              EMT312
3633  004674  073620                              EHT1
3634  004676  073744                              EDT1
3635  004700  074034                              EFT1
3636
3637
3638                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
3639  004702  073104                              EMT313
3640  004704  073620                              EHT1
3641  004706  073744                              EDT1
3642  004710  074034                              EFT1
3643
3644
3645                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
3646  004712  073124                              EMT314
3647  004714  073620                              EHT1
3648  004716  073744                              EDT1
3649  004720  074034                              EFT1
3650
3651
3652                                    ;ERROR  315     WRITE LOCK ERROR
3653  004722  073136                              EMT315
3654  004724  073620                              EHT1
3655  004726  073744                              EDT1
3656  004730  074034                              EFT1
3657
3658
3659                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
3660  004732  073150                              EMT316
3661  004734  073620                              EHT1
3662  004736  073744                              EDT1
3663  004740  074034                              EFT1
3664
```

```
3665
3666                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
3667  004742  073162                              EMT317
3668  004744  073620                              EHT1
3669  004746  073744                              EDT1
3670  004750  074034                              EFT1
3671
3672
3673                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
3674  004752  073172                              EMT320
3675  004754  073620                              EHT1
3676  004756  073744                              EDT1
3677  004760  074034                              EFT1
3678
3679
3680                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
3681  004762  073202                              EMT321
3682  004764  073620                              EHT1
3683  004766  073744                              EDT1
3684  004770  074034                              EFT1
3685
3686
3687                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
3688  004772  073212                              EMT322
3689  004774  073620                              EHT1
3690  004776  073744                              EDT1
3691  005000  074034                              EFT1
3692
3693
3694                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
3695  005002  073220                              EMT323
3696  005004  073620                              EHT1
3697  005006  073744                              EDT1
3698  005010  074034                              EFT1
3699
3700
3701                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3702  005012  073230                              EMT324
3703  005014  073620                              EHT1
3704  005016  073744                              EDT1
3705  005020  074034                              EFT1
3706
3707
3708                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3709  005022  073242                              EMT325
3710  005024  073620                              EHT1
3711  005026  073744                              EDT1
3712  005030  074034                              EFT1
3713
3714
3715                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
3716  005032  073254                              EMT326
3717  005034  073620                              EHT1
3718  005036  073744                              EDT1
3719  005040  074034                              EFT1
3720
```

```
3721
3722                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
3723  005042  073272                              EMT327
3724  005044  073620                              EHT1
3725  005046  073744                              EDT1
3726  005050  074034                              EFT1
3727
3728
3729                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3730  005052  073304                              EMT330
3731  005054  073620                              EHT1
3732  005056  073744                              EDT1
3733  005060  074034                              EFT1
3734
3735
3736                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3737  005062  073314                              EMT331
3738  005064  073620                              EHT1
3739  005066  073744                              EDT1
3740  005070  074034                              EFT1
3741
3742
3743                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3744  005072  073326                              EMT332
3745  005074  073620                              EHT1
3746  005076  073744                              EDT1
3747  005100  074034                              EFT1
3748
3749
3750                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3751  005102  073340                              EMT333
3752  005104  073620                              EHT1
3753  005106  073744                              EDT1
3754  005110  074034                              EFT1
3755
3756
3757                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3758  005112  073356                              EMT334
3759  005114  073620                              EHT1
3760  005116  073744                              EDT1
3761  005120  074034                              EFT1
3762
3763
3764                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3765  005122  073374                              EMT335
3766  005124  073620                              EHT1
3767  005126  073744                              EDT1
3768  005130  074034                              EFT1
3769
3770
3771                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3772  005132  073414                              EMT336
3773  005134  073[illegible]                       EHT336
3774  005136  074000                              EDT336
3775  005140  074070                              EFT336
3776
```

```
                                ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
005142  073424                          EMT337
005144  073714                          EHT337
005146  074010                          EDT337
005150  074100                          EFT337


                                ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
005152  073434                          EMT340
005154  073702                          EHT336
005156  074000                          EDT336
005160  074070                          EFT336


                                ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
005162  073446                          EMT341
005164  073702                          EHT336
005166  074000                          EDT336
005170  074070                          EFT336


                                ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
005172  073454                          EMT342
005174  073620                          EHT1
005176  073744                          EDT1
005200  074034                          EFT1


                                ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
005202  073466                          EMT343
005204  073620                          EHT1
005206  073744                          EDT1
005210  074034                          EFT1


                                ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
005212  073500                          EMT344
005214  073726                          EHT344
005216  074020                          EDT344
005220  074110                          EFT344


                                ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
005222  073512                          EMT345
005224  073620                          EHT1
005226  073744                          EDT1
005230  074034                          EFT1


                                ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
005232  073522                          EMT346
005234  073620                          EHT1
005236  073744                          EDT1
005240  074034                          EFT1
```

```
3833
3834                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3835  005242  073536                            EMT347
3836  005244  073620                            EHT1
3837  005246  073744                            EDT1
3838  005250  074034                            EFT1
3839
3840
3841                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3842  005252  073550                            EMT350
3843  005254  073620                            EHT1
3844  005256  073744                            EDT1
3845  005260  074034                            EFT1
3846
3847
3848                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3849  005262  073566                            EMT351
3850  005264  073620                            EHT1
3851  005266  073744                            EDT1
3852  005270  074034                            EFT1
3853
3854
3855                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3856  005272  073576                            EMT352
3857  005274  000000                            0
3858  005276  000000                            0
3859  005300  000000                            0
3860
3861
3862                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3863  005302  073602                            EMT353
3864  005304  073740                            EHT353
3865  005306  074032                            EDT353
3866  005310  074120                            EFT353
3867
3868
3869                                    ;ERROR  354     BSE SHOULD NOT BE SET
3870  005312  073612                            EMT354
3871  005314  073620                            EHT1
3872  005316  073744                            EDT1
3873  005320  074034                            EFT1
3874
3875
3876                                    ;PUT ERROR TABLE HERE
3877                                    .SBTTL  ERROR TABLE USAGE
3878
3879                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3880                                    ;NUMBER, I.E.,
3881                                    ;
3882                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3883                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3884                                    ;       EDT - ERROR DATA TABLE ADDRESS
3885                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3886                                    ;
3887                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3888                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3889                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3890                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3891                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3892                    ;
3893                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3894                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3895                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3896                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3897                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3898                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3899                    ;MUST ALSO HAVE A FORMAT.
3900                    ;
3901                    ;IN SUMMARY,
3902                    ;
3903                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3904                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3905                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3906                    ;
3907
```

```
                                        .SBTTL  START OF PROGRAM


005322                                  START:

                                        ;CLEAR AND SETUP FOR TEST EXECUTION
005322  000240                                 NOP
005324  000005                                 RESET                   ;INITIALIZE THE SYSTEM
005326  013746  000300                         MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
005332  012746  005340                         MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
005336  000002                                 RTI                     ;;POP NEW PC AND PS
005340                                  64$:

                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
005340  012706  001114                         MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
005344  005026                                 CLR     (R6)+           ;;CLEAR MEMORY LOCATION
005346  022706  001154                         CMP     #$WR,R6 ;;DONE?
005352  001374                                 BNE     .-6             ;;LOOP BACK IF NO
005354  012706  001100                         MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
005360  012737  063032  000020                 MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005366  012737  000340  000022                 MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005374  012737  063534  000030                 MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005402  012737  000340  000032                 MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005410  012737  065274  000034                 MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005416  012737  000340  000036                 MOV     #340,@#TRAPVEC+2;LEVEL 7
005424  012737  065402  000024                 MOV     #$PWRUP,@#PWRVEC ;;POWER FAILURE VECTOR
005432  012737  000340  000026                 MOV     #340,@#PWRVEC+2 ;;LEVEL 7
005440  013737  042070  042062                 MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
005446  005037  001206                         CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
005452  005037  001210                         CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
005456  112737  000001  001131                 MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
005464  012737  005464  001122                 MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005472  012737  005472  001124                 MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
005500  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005504  012737  005540  000004                 MOV     #65$,@#ERRVEC   ;;SET UP ERROR VECTOR
005512  012737  177570  001154                 MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005520  012737  177570  001156                 MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005526  022777  177777  173420                 CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005534  001012                                 BNE     67$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                       ;;AND  THE HARDWARE SWR IS NOT = -1
005536  000403                                 BR      66$             ;;BRANCH IF NO TIMEOUT
005540  012716  005546          65$:           MOV     #66$,(SP)       ;;SET UP FOR TRAP RETURN
005544  000002                                 RTI
005546  012737  000176  001154  66$:           MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005554  012737  000174  001156                 MOV     #DISPREG,DISPLAY
005562  012637  000004          67$:           MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

005566  005037  001230                         CLR     $PASS           ;;CLEAR PASS COUNT
005572  132737  000200  001243                 BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005600  001403                                 BEQ     68$             ;;YES,USE NON-APT SWITCH
005602  012737  001244  001154                 MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005610                                  68$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005610  005227  177777                          INC     #-1             ;;FIRST TIME?
005614  001053                                  BNE     69$             ;;BRANCH IF NO
005616  022737  042224  000042                 CMP     #$ENDAD,@#42    ;;ACT-11?
005624  001447                                  BEQ     69$             ;;BRANCH IF YES
005626  104401  005674                          TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005632  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005636  001012                                  BNE     71$             ;;BRANCH IF YES
005640  123727  001242  000001                 CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005646  001406                                  BEQ     71$             ;;BRANCH IF YES
005650  023727  001154  000176                 CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005656  001005                                  BNE     72$             ;;BRANCH IF NO
005660  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
005662  000403                                  BR      72$
005664  112737  000001  001150  71$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005672                                  72$:
005672  000424                          BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@DZRMCA - RM03 FUNCTIONAL TEST, PART 1@<CRLF>
005744                                  69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005744  005737  000042                          TST     42              ;IS LOC 42 ZERO ??
005750  001003                                  BNE     10$             ;NO - NOT IN STANDALONE
005752  105737  001242                          TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
005756  001411                                  BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
005760                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
005760  132737  000200  001243                 BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
005766  001003                                  BNE     20$             ;NO
005770  012737  000377  001300                 MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
005776                                  20$:


                                        ;GO TO COMMON START CODE
005776  000137  006620                          JMP     CMNSTART
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
006002
006002  004737  063744          STANDALONE:
                                        JSR     PC,$TKINT       ;INITIALIZE CONSOLE

                                ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
006006  005327  000001                  DEC     #1              ;FIRST START ??
006012  100024                          BPL     10$             ;YES !!


                                ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
006014  104401  066276          5$:     TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
006020  104411                          RDCHR                   ;GET RESPONSE
006022  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
006026  104401  001176                  TYPE    ,$TMP1
006032  123727  001176  000131          CMPB    $TMP1,#'Y       ;YES RESPONSE??
006040  001002                          BNE     6$              ;NO!!
006042  000137  006752                  JMP     READY           ;KEEP PREVIOUS PARAMETERS
006046  123727  001176  000116  6$:     CMPB    $TMP1,#'N       ;NO RESPONSE??
006054  001420                          BEQ     20$             ;GET NEW PARAMETERS
006056  104401  066145                  TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
006062  000754                          BR      5$              ;RETRY
006064                          10$:

                                ;SEE IF OPERATOR WANTS HELP FILE
006064  104401  066147                  TYPE    ,HELPQST        ;WANT HELP ??
006070  104411                          RDCHR                   ;GET RESPONSE
006072  012637  001176                  MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006076  104401  001176                  TYPE    ,$TMP1
006102  123727  001176  000131          CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
006110  001002                          BNE     20$             ;NO - DONT TYPE HELP
006112  104401  105642                  TYPE    ,HELP           ;YES - TYPE HELP TEXT
006116                          20$:

                                ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
006116  104401  066203                  TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
006122  104411                          RDCHR                   ;GET RESPONSE
006124  012637  001176                  MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006130  104401  001176                  TYPE    ,$TMP1
006134  123727  001176  000131          CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
006142  001137                          BNE     90$             ;NO !!
006144                          30$:

                                ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
006144  104401  066335                  TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
006150  013746  001276                  MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
006154  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006156  104401  066136                  TYPE    ,CLSPRN
006162  104413                          RDOCT                   ;GET NEW BUS ADDRESS
006164  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
006170  001412                          BEQ     50$             ;YES-SKIP TO NEXT ENTRY
006172  022737  160000  001176          CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
006200  101403                          BLOS    40$             ;YES
006202  104401  066362                  TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
006206  000756                          BR      30$             ;RETRY
006210  013737  001176  001276  40$:     MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
006216  113737  001272  001176  50$:     MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
006224  105037  001177                  CLRB    $TMP1+1
```

```
4058  006230  104401  066443                  TYPE    ,CNSL03
4059  006234  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
4060  006240  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
4061  006242     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
4062  006243     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
4063  006244  104401  066136                  TYPE    ,CLSPRN
4064  006250  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
4065  006252  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
4066  006256  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
4067  006260  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
4068  006266  101003                          BHI     60$             ;YES!!
4069  006270  104401  066473                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
4070  006274  000750                          BR      50$             ;RETRY
4071  006276  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
4072  006304  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
4073  006312  006237  001176                  ASR     $TMP1
4074  006316  006237  001176                  ASR     $TMP1
4075  006322  006237  001176                  ASR     $TMP1
4076  006326  006237  001176                  ASR     $TMP1
4077  006332  006237  001176                  ASR     $TMP1
4078  006336  105037  001177                  CLRB    $TMP1+1
4079  006342  104401  066547                  TYPE    ,CNSL05
4080  006346  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
4081  006352  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
4082  006354     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
4083  006355     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
4084  006356  104401  066136                  TYPE    ,CLSPRi
4085  006362  104413                          RDOCT                   ;GET NEW PRIORITY
4086  006364  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
4087  006370  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
4088  006372  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
4089  006400  002403                          BLT     80$             ;YES!!
4090  006402  104401  066603                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
4091  006406  000736                          BR      70$             ;RETRY
4092  006410                          80$:                            ;STORE NEW PRIORITY
4093  006410  006337  001176                  ASL     $TMP1
4094  006414  006337  001176                  ASL     $TMP1
4095  006420  006337  001176                  ASL     $TMP1
4096  006424  006337  001176                  ASL     $TMP1
4097  006430  006337  001176                  ASL     $TMP1
4098  006434  113737  001176  001273          MOVB    $TMP1,$VECT1+1
4099  006442                          90$:
4100
4101                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
4102  006442  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
4103  006446  104401  066630                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
4104  006452  104401  066141                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
4105  006456  104411                          RDCHR                   ;GET RESPONSE
4106  006460  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
4107  006464  104401  001176                  TYPE    ,$TMP1
4108  006470  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
4109  006476  001017                          BNE     110$            ;NO
4110  006500  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
4111  006506  000444                          BR      140$            ;SKIP TO NEXT ENTRY
4112  006510  104401  066141          100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
4113  006514  104411                          RDCHR                   ;GET RESPONSE
```

```
 4114  006516  012637  001176            MOV     (SP)+,$TMP1     ;ECHO RESPONSE
 4115  006522  104401  001176            TYPE    ,$TMP1
 4116  006526  023727  001176  000015    CMP     $TMP1,#CR       ;CARRIAGE RETURN??
 4117  006534  001431                    BEQ     140$
 4118  006536  023727  001176  000060  110$:  CMP     $TMP1,#'0       ;NUMBER < 0??
 4119  006544  002404                    BLT     120$            ;YES!!
 4120  006546  023727  001176  000067    CMP     $TMP1,#'7       ;NUMBER > 7??
 4121  006554  003403                    BLE     130$            ;NO!!
 4122  006556  104401  066145        120$:  TYPE    ,QSTMRK         ;TYPE "?"
 4123  006562  000752                    BR      100$            ;RETRY
 4124  006564  013701  001176        130$:  MOV     $TMP1,R1        ;R1=DRIVE NUMBER
 4125  006570  042701  177770            BIC     #^C7,R1
 4126  006574  116102  067070            MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
 4127  006600  042702  177400            BIC     #^C377,R2               ;CLEAR UNUSED BITS
 4128  006604  050237  001300            BIS     R2,$DEVM                ;SET DEVICE # IN MAP
 4129  006610  122737  000377  001300    CMPB    #377,$DEVM      ;DONE ??
 4130  006616  101334                    BHI     100$            ;NO
 4131  006620                        140$:
 4132
```

```
                                        GET VALUE FOR SOFTWARE SWITCH REGISTER

 4133  006620                           CMNSTART:
 4134
 4135                                   ;ASSEMBLE TEST QUE FROM DEVICE MAP
 4136  006620  013700  001300                   MOV     $DEVM,R0        ;R0 = DEVICE MAP
 4137  006624  012701  001450                   MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
 4138  006630  010137  001446                   MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
 4139  006634  012702  000001                   MOV     #1,R2           ;R2 = DEVICE POINTER
 4140  006640  005003                           CLR     R3              ;R3 = DEVICE NUMBER
 4141  006642  030200                   10$:    BIT     R2,R0           ;IS THIS DEVICE IN MAP ??
 4142  006644  001406                           BEQ     20$             ;NO !!
 4143  006646  010311                           MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
 4144  006650  116361  067070  000001           MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
 4145  006656  062701  000002                   ADD     #2,R1           ;ADVANCE ENTRY POINTER
 4146  006662  006302                   20$:    ASL     R2              ;ADVANCE DEVICE POINTER
 4147  006664  105702                           TSTB    R2              ;DONE ALL DEVICES ??
 4148  006666  001402                           BEQ     25$             ;YES
 4149  006670  005203                           INC     R3              ;ADVANCE DEVICE NUMBER
 4150  006672  000763                           BR      10$             ;ENTER NEXT DEVICE
 4151  006674  005011                   25$:    CLR     (R1)            ;TERMINATE TEST QUE
 4152
 4153                                   ;SIZE FOR CLOCK
 4154  006676  004737  044534                   JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
 4155  006702  000403                           BR      40$             ;YES - CLOCK IS PRESENT
 4156  006704  104000                   30$:    ERROR                   ;NO CLOCK
 4157  006706  000000                           HALT
 4158  006710  000775                           BR      30$
 4159  006712                           40$:
 4160                                   .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 4161  006712  005737  000042                   TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
 4162  006716  001012                           BNE     64$             ;;BRANCH IF YES
 4163  006720  123727  001242  000001           CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
 4164  006726  001406                           BEQ     64$             ;;BRANCH IF YES
 4165  006730  023727  001154  000176           CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
 4166  006736  001005                           BNE     65$             ;;BRANCH IF NO
 4167  006740  104407                           GTSWR                   ;;GET SOFT-SWR SETTINGS
 4168  006742  000403                           BR      65$
 4169  006744  112737  000001  001150   64$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
 4170  006752                           65$:
 4171  006752  000240                   READY:  NOP                     ;READY TO START TEST
 4172  006754  004737  063744                   JSR     PC,$TKINT       ;INITIALIZE TTY
 4173  006760  013746  000300                   MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
 4174  006764  012746  006772                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 4175  006770  000002                           RTI                     ;;POP NEW PC AND PS
 4176  006772                           64$:
 4177  006772  117737  172450  001234           MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
 4178  007000  005037  001472                   CLR     MEDENB          ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
007004                          TST1:
007004  000240                          NOP                             ;START OF TEST
007006  012737  007022  001122          MOV     #1$,$LPAOR
007014  012737  007022  001124          MOV     #1$,$LPERR
007022                          1$:
007022  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007026  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007032  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007036  012737  000001  001226          MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
007044  005001                          CLR     R1
007046  013746  000004                  MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
007052  013746  000006                  MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
007056  012737  007150  000004          MOV     #3$,ERRVEC
007064  012737  000300  000006          MOV     #PR6,ERRVEC+2
                                
007072  110160  000001                  MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
007076  010160  000002                  MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
007102  016002  000002                  MOV     RMWC(R0),R2
007106  010160  000004                  MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
007112  016002  000004                  MOV     RMBA(R0),R2
007116  010160  000010                  MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
007122  016002  000010                  MOV     RMCS2(R0),R2
007126  010160  000022                  MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
007132  016002  000022                  MOV     RMDB(R0),R2
007136  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007142  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007146  000415                          BR      7$                      ;NO BUS TIMEOUT OCCURRED
                                
007150  022626                  3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
007152  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007156  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007162  104110                          ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
007164  005737  000042                  TST     42                      ;STAND ALONE MODE??
007170  001002                          BNE     5$                      ;NO!!
007172  000137  005322                  JMP     START                   ;YES-GO GET $BASE
007176  000137  042034          5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
007202                          7$:
                                
                                ;;*****************************************************************
                                ;*TEST 2         DEVICE AVAILABLE TEST
                                
                                ;*****************************************************************
007202                          TST2:
007202  000004                          SCOPE                           ;SCOPE CALL
007204  000240                          NOP                             ;START OF TEST
007206  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007212  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007216  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007222  012737  000002  001226          MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
                                
007230  004737  053210                  JSR     PC,CNTCLR
007234  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007236  000240                          NOP                             ;RETURN HERE IF ERROR
```

```
4235  007240  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4236  007242  000137  007362                  JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
4237  007246                          2$:
4238  007246  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
4239  007252  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
4240  007256  012737  007346  000004          MOV     #5$,ERRVEC
4241  007264  013737  000300  000006          MOV     PR6,ERRVEC+2
4242
4243  007272  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
4244  007300  016037  000010  001174          MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
4245  007306  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
4246  007312  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
4247  007316  032737  010000  001174          BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
4248  007324  001402                          BEQ     3$              ;NO!!
4249  007326  104111                          ERROR   111             ;NONEXISTENT DEVICE
4250  007330  000414                          BR      7$
4251  007332  032737  004000  001176  3$:     BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
4252  007340  001012                          BNE     9$              ;YES!!
4253  007342  104112                          ERROR   112             ;DEVICE NOT AVAILABLE
4254  007344  000406                          BR      7$
4255
4256  007346  022626                  5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
4257  007350  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
4258  007354  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
4259  007360  104113                          ERROR   113             ;BUS TIMEOUT (04 TRAP)
4260  007362  000137  042000          7$:     JMP     $EOSP
4261
4262  007366                          9$:
4263
4264                                  ;;*****************************************************************
4265                                  ;*TEST 3        DRIVE TYPE TEST
4266
4267                                  ;;*****************************************************************
4268  007366                          TST3:
4269  007366  000004                          SCOPE                   ;SCOPE CALL
4270  007370  000240                          NOP                     ;START OF TEST
4271  007372  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4272  007376  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4273  007402  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4274  007406  012737  000003  001226          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4275
4276  007414  004737  053210                  JSR     PC,CNTCLR
4277  007420  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
4278  007422  000240                          NOP                     ;RETURN HERE IF ERROR
4279  007424  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4280  007426  000137  007524                  JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
4281  007432                          2$:
4282  007432  112737  000026  001504          MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
4283  007440  112737  000200  001505          MOVB    #200,GETINX+1
4284  007446  012737  007530  001354          MOV     #5$,RMDTI       ;RMDT INPUT BUFFER = 5$
4285  007454  004737  044046                  JSR     PC,GET          ;GO GET DRIVE TYPE
4286  007460  000402                          BR      3$      ;GO TO 3$ IF NO ERROR
4287  007462  000240                          NOP                     ;RETURN HERE IF ERROR
4288  007464  104000                          ERROR                   ;ERROR NUMBER SPECIFIED BY GET
4289  007466  022737  020024  001354  3$:     CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
4290  007474  001415                          BEQ     5$              ;YES!!
```

```
                                T3      DRIVE TYPE TEST

4291  007476  022737  024024  001354            CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
4292  007504  001411                            BEQ     5$              ;YES!!
4293  007506  012737  020024  001176            MOV     #SNGPRT,$TMP1
4294  007514  012737  024024  001200            MOV     #DULPRT,$TMP2
4295  007522  104114                            ERROR   114             ;NOT AN RM03
4296  007524  000137  042000            4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
4297
4298  007530                            5$:
4299                                    ;;**********************************************************************
4300                                    ;*TEST 4         UNIBUS INITIALIZE TEST
4301
4302                                    ;;**********************************************************************
4303  007530                            TST4:
4304  007530  000004                            SCOPE                   ;SCOPE CALL
4305  007532  000240                            NOP                     ;START OF TEST
4306  007534  012737  000001  001206            MOV     #1,$TIMES       ;LOAD ITERATION COUNT
4307  007542  012737  007556  001122            MOV     #1$,$LPADR
4308  007550  012737  007556  001124            MOV     #1$,$LPERR
4309  007556                            1$:
4310  007556  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4311  007562  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4312  007566  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4313  007572  012737  000004  001226            MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4314
4315
4316  007600  004737  053210                    JSR     PC,CNTCLR
4317  007604  000404                            BR      10$     ;GO TO 10$ IF NO ERROR
4318  007606  000240                            NOP                     ;RETURN HERE IF ERROR
4319  007610  104000                            ERROR                   ;ERROR NUMBER DEFINED BY SUB
4320  007612  000137  010414                    JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4321  007616  012702  000101            10$:    MOV     #65.,R2         ;SET OR AND RESET IR
4322  007622  012737  000000  001420            MOV     #0,RMDB0        ;WRITE ZEROS IN DATA SILO
4323  007630  112737  000022  001533            MOVB    #RMDB,PUTINX    ;SETUP REGISTER INDEX
4324  007636  112737  000200  001534            MOVB    #200,PUTINX+1
4325  007644                            20$:
4326  007644  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4327  007650  000404                            BR      30$     ;GO TO 30$ IF NO ERROR
4328  007652  000240                            NOP                     ;RETURN HERE IF ERROR
4329  007654  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
4330  007656  000137  010414                    JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4331  007662  005302                    30$:    DEC     R2              ;DECREMENT COUNT
4332  007664  001367                            BNE     20$             ;WRITE SILO AGAIN
4333  007666  012737  177777  001402            MOV     #-1,RMBA0       ;RMBA=ALL ONES
4334  007674  012737  177777  001412            MOV     #-1,RMER10      ;RMER1=ALL ONES
4335  007702  012737  177777  001440            MOV     #-1,RMER20      ;RMER2 = ALL ONES
4336  007710  012737  040001  001422            MOV     #DMD!DBEN,RMMR10        ;SET DIAGNOSTIC MODE
4337  007716  012737  003577  001376            MOV     #003577,RMCS10
4338  007724  012737  021037  001406            MOV     #021037,RMCS20
4339
4340  007732  012702  001533                    MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
4341  007736  112722  000004                    MOVB    #RMBA,(R2)+
4342  007742  112722  000014                    MOVB    #RMER1,(R2)+
4343  007746  112722  000042                    MOVB    #RMER2,(R2)+
4344  007752  112722  000024                    MOVB    #RMMR1,(R2)+
4345  007756  112722  000000                    MOVB    #RMCS1,(R2)+
4346  007762  112722  000010                    MOVB    #RMCS2,(R2)+
```

```
4347  007766  112722  000200                  MOVB    #200,(R2)+
4348  007772  004737  044316                  JSR     PC,PUT              ;GO WRITE REGISTERS VIA PUT SUB
4349  007776  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4350  010000  000240                          NOP                         ;RETURN HERE IF ERROR
4351  010002  104000                          ERROR                       ;ERROR DEFINED BY PUT SUB
4352  010004  000137  010414                  JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4353  010010  000005                  40$:    RESET                       ;UNIBUS INITIALIZE
4354  010012  004737  063744                  JSR     PC,$TKINT           ;INITIALIZE CONSOLE
4355  010016  111160  000010                  MOVB    (R1),RMCS2(R0)      ;SELECT DEVICE
4356  010022  004737  043762                  JSR     PC,GETSTS           ;GO SET UP FOR STATUS FETCH
4357  010026  004737  044046                  JSR     PC,GET              ;GO READ REGISTERS VIA GET SUB
4358  010032  000403                          BR      50$     ;GO TO 50$ IF NO ERROR
4359  010034  000240                          NOP                         ;RETURN HERE IF ERROR
4360  010036  104000                          ERROR                       ;ERROR DEFINED BY GET SUB
4361  010040  000565                          BR      220$                ;SKIP REMAINDER OF TEST
4362
4363  010042  013737  001326  001142  50$:    MOV     RMCS1I,$BDDAT       ;VERIFY RMCS1
4364  010050  042737  100000  001142          BIC     #SC,$BDDAT          ;IGNORE SPECIAL CONDITION
4365  010056  012737  004200  001140          MOV     #DVA!RDY,$GDDAT     ;EXPECT DVA & RDY
4366  010064  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED, RECEIVED
4367  010072  001401                          BEQ     60$                 ;BRANCH IF EQUAL
4368  010074  104115                          ERROR   115                 ;RMCS1 NOT INITIALIZED
4369
4370  010076  005037  001140          60$:    CLR     $GDDAT              ;VERIFY RMBA IS ZERO
4371  010102  013737  001332  001142          MOV     RMBAI,$BDDAT
4372  010110  001401                          BEQ     70$                 ;BRANCH IF ZERO
4373  010112  104116                          ERROR   116                 ;RMBA NOT INITIALIZED
4374
4375  010114  013737  001336  001142  70$:    MOV     RMCS2I,$BDDAT       ;VERIFY RMCS2
4376  010122  005046                          CLR     -(SP)               ;EXPECT IR & UNIT NUMBER
4377  010124  111116                          MOVB    (R1),(SP)
4378  010126  052716  000100                  BIS     #IR,(SP)
4379  010132  012637  001140                  MOV     (SP)+,$GDDAT
4380  010136  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
4381  010144  001401                          BEQ     90$                 ;BRANCH IF EQUAL
4382  010146  104117                          ERROR   117                 ;RMCS2 NOT INITIALIZED
4383
4384  010150  005037  001140          90$:    CLR     $GDDAT              ;VERIFY RMER1
4385  010154  013737  001342  001142          MOV     RMER1I,$BDDAT
4386  010162  042737  040000  001142          BIC     #UNS,$BDDAT         ;IGNORE UNSAFE
4387  010170  001401                          BEQ     110$                ;BRANCH IF ZERO
4388  010172  104120                          ERROR   120                 ;RMER1 NOT INITIALIZED
4389
4390  010174  013737  001344  001142  110$:   MOV     RMASI,$BDDAT        ;VERIFY RMAS
4391  010202  005002                          CLR     R2                  ;CLEAR ALL BUT THIS
4392  010204  116102  000001                  MOVB    1(R1),R2            ;DRIVES ATTENTION BIT
4393  010210  000302                          SWAB    R2
4394  010212  005102                          COM     R2
4395  010214  000240                          NOP
4396  010216  040237  001142                  BIC     R2,$BDDAT
4397  010222  001401                          BEQ     130$                ;BRANCH IF ITS 0
4398  010224  104121                          ERROR   121                 ;RMAS NOT INITIALIZED
4399
4400  010226  013737  001352  001142  130$:   MOV     RMMR1I,$BDDAT       ;VERIFY RMMR
4401  010234  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT           ;IGNORE WORD CLOCK, SCT, TRK
4402  010242  012737  000010  001140          MOV     #MWD,$GDDAT         ;EXPECT WRITE DATA BIT
```

```
4403  010250  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
4404  010256  001401                          BEQ     170$            ;BRANCH IF 0
4405  010260  104122                          ERROR   122             ;RMMR NOT INITIALIZED
4406
4407  010262  005037  001140          170$:   CLR     $GDDAT          ;EXPECT ZEROS
4408  010266  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
4409  010274  001401                          BEQ     190$
4410  010276  104124                          ERROR   124             ;RMEC2 NOT INITIALIZED
4411
4412  010300  013737  001366  001142  190$:   MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
4413  010306  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
4414  010314  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST,TAG BIT ON
4415  010322  023737  001140  001142          CMP     $GDDAT,$BDDAT
4416  010330  001401                          BEQ     210$
4417  010332  104125                          ERROR   125             ;RMER2 NOT INITIALIZED
4418
4419  010334  005037  001140          210$:   CLR     $GDDAT          ;EXPECT ALL ZEROS
4420  010340  013737  001370  001142          MOV     RMER2I,$BDDAT   ;VERIFY RMER2
4421  010346  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
4422  010354  001401                          BEQ     215$            ;BRANCH IF OTHER BITS 0
4423  010356  104173                          ERROR   173
4424  010360  013737  001340  001142  215$:   MOV     RMDSI,$BDDAT    ;CHECK DRIVE STATUS REGISTER
4425  010366  042737  177177  001142          BIC     #^C<DPR!DRY>,$BDDAT
4426  010374  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED STATUS
4427  010402  023737  001142  001140          CMP     $BDDAT,$GDDAT   ;COMPARE EXPECTED & RECEIVED STATUS
4428  010410  001401                          BEQ     220$            ;DRIVE STATUS IS OK
4429  010412  104123                          ERROR   123             ;REPORT ERROR IN DRIVE STATUS
4430  010414                          220$:
4431
4432
4433                                  ;;*****************************************************************
4434                                  ;*TEST 5         CONTROLLER CLEAR TEST
4435
4436                                  ;;*****************************************************************
4437  010414                          TST5:
4438  010414  000004                          SCOPE                   ;SCOPE CALL
4439  010416  000240                          NOP                     ;START OF TEST
4440  010420  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4441  010424  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4442  010430  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4443  010434  012737  000005  001226          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4444
4445  010442  004737  053210                  JSR     PC,CNTCLR
4446  010446  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4447  010450  000240                          NOP                     ;RETURN HERE IF ERROR
4448  010452  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4449  010454  000137  010752                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4450  010460  012702  000101          10$:    MOV     #65.,R2
4451  010464  012737  000000  001420          MOV     #0,RMDB0                ;WRITE ZEROS IN DATA SILO
4452  010472  112737  000022  001533          MOVB    #RMDB,PUTINX    ;SETUP REGISTER INDEX TABLE
4453  010500  112737  000200  001534          MOVB    #200,PUTINX+1
4454  010506                          20$:
4455  010506  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4456  010512  000404                          BR      30$     ;GO TO 30$ IF NO ERROR
4457  010514  000240                          NOP                     ;RETURN HERE IF ERROR
4458  010516  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
```

```
4459  010520  000137  010752                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4460  010524  005302                    30$:    DEC     R2              ;DECREMENT COUNT
4461  010526  001367                            BNE     20$             ;AND WRITE SILO AGAIN IF NOT DONE
4462  010530  012737  177777  001402            MOV     #-1,RMBA0
4463  010536  012737  177777  001412            MOV     #-1,RMER10
4464  010544  012737  177777  001440            MOV     #-1,RMER20
4465  010552  012737  040001  001422            MOV     #DMD!DBEN,RMMR10
4466  010560  012737  003577  001376            MOV     #003577,RMCS10
4467  010566  012737  021037  001406            MOV     #021037,RMCS20
4468  010574  012702  001533                    MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
4469  010600  112722  000004                    MOVB    #RMBA,(R2)+
4470  010604  112722  000014                    MOVB    #RMER1,(R2)+
4471  010610  112722  000042                    MOVB    #RMER2,(R2)+
4472  010614  112722  000024                    MOVB    #RMMR1,(R2)+
4473  010620  112722  000000                    MOVB    #RMCS1,(R2)+
4474  010624  112722  000010                    MOVB    #RMCS2,(R2)+
4475  010630  112722  000200                    MOVB    #200,(R2)+
4476  010634  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4477  010640  000403                            BR      40$     ;GO TO 40$ IF NO ERROR
4478  010642  000240                            NOP                     ;RETURN HERE IF ERROR
4479  010644  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
4480  010646  000430                            BR      70$
4481
4482  010650                            40$:
4483  010650  004737  053210                    JSR     PC,CNTCLR
4484  010654  000404                            BR      50$     ;GO TO 50$ IF NO ERROR
4485  010656  000240                            NOP                     ;RETURN HERE IF ERROR
4486  010660  104000                            ERROR                   ;ERROR NUMBER DEFINED BY SUB
4487  010662  000137  010752                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4488  010666  004737  043762            50$:    JSR     PC,GETSTS
4489  010672  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4490  010676  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
4491  010700  000240                            NOP                     ;RETURN HERE IF ERROR
4492  010702  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
4493  010704  000137  010752                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4494  010710                            60$:
4495  010710  004737  053326                    JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
4496  010714  000405                            BR      70$             ;GO TO 70$ IF NO ERROR
4497  010716  000240                            NOP                     ;RETURN HERE IF ERROR
4498  010720  104000                            ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
4499  010722  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4500  010724  000137  010752                    JMP     80$             ;GO TO 80$ IF ERROR WAS FOUND
4501  010730                            70$:
4502  010730  012737  000000  001376            MOV     #NOP,RMCS10     ;CHANGE  FUNCTION CODE FOR ERROR CHECK
4503  010736  004737  045674                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4504  010742  000403                            BR      80$             ;GO TO 80$ IF NO ERROR
4505  010744  000240                            NOP                     ;RETURN HERE IF ERROR
4506  010746  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4507  010750  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4508  010752                            80$:
4509
4510                                    ;;*****************************************************************
4511                                    ;*TEST 6         ERROR CLEAR TEST
4512
4513                                    ;;*****************************************************************
4514  010752                            TST6:
```

```
4515  010752  000004                          SCOPE                   ;SCOPE CALL
4516  010754  000240                          NOP                     ;START OF TEST
4517  010756  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4518  010762  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4519  010766  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4520  010772  012737  000006  001226          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4521
4522  011000  004737  053210                  JSR     PC,CNTCLR
4523  011004  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4524  011006  000240                          NOP                     ;RETURN HERE IF ERROR
4525  011010  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4526  011012  000137  011156                  JMP     50$     ;GO TO 50$ IF ERROR WAS FOUND
4527  011016                          10$:
4528  011016  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4529  011024  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4530  011032  012737  040000  001376          MOV     #TRE,RMCS10     ;RMCS1 OUTPUT BUFFER = TRE
4531  011040  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
4532  011044  000403                          BR      20$     ;GO TO 20$ IF NO ERROR
4533  011046  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
4534  011050  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4535  011052  000441                          BR      50$             ;SKIP REST OF TEST
4536  011054  112737  000000  001504  20$:     MOVB    #RMCS1,GETINX
4537  011062  112737  000010  001505          MOVB    #RMCS2,GETINX+1
4538  011070  112737  000200  001506          MOVB    #200,GETINX+2
4539  011076  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4540  011102  000403                          BR      30$     ;GO TO 30$ IF NO ERROR
4541  011104  000240                          NOP                     ;RETURN HERE IF ERROR
4542  011106  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4543  011110  000422                          BR      50$             ;SKIP REST OF TEST
4544  011112  013737  001326  001142  30$:     MOV     RMCS1I,$BDDAT   ;CHECK TRE & MCPE
4545  011120  005037  001140                  CLR     $GDDAT          ;EXPECT 0'S
4546  011124  042737  117777  001142          BIC     #^C<TRE!MCPE>,$BDDAT
4547  011132  001401                          BEQ     40$             ;BRANCH IF TRE & MCPE = 0
4548  011134  104137                          ERROR   137             ;MCPE, OR TRE NOT CLEARED
4549
4550  011136  013737  001336  001142  40$:     MOV     RMCS2I,$BDDAT   ;CHECK RMCS2
4551  011144  042737  104377  001142          BIC     #^C<WCE!UPE!NED!PGE!MXF!MDPE>,$BDDAT
4552  011152  001401                          BEQ     50$
4553  011154  104140                          ERROR   140             ;RMCS2 NOT CLEARED
4554
4555  011156                          50$:
4556
4557                                  ;;*****************************************************************
4558                                  ;*TEST 7        DRIVE STATUS TEST
4559
4560                                  ;;*****************************************************************
4561  011156                          TST7:
4562  011156  000004                          SCOPE                   ;SCOPE CALL
4563  011160  000240                          NOP                     ;START OF TEST
4564  011162  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4565  011166  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4566  011172  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4567  011176  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4568
4569  011204  004737  053210                  JSR     PC,CNTCLR
4570  011210  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
```

```
4571  011212  000240                          NOP                         ;RETURN HERE IF ERROR
4572  011214  104000                          ERROR                       ;ERROR NUMBER DEFINED BY SUB
4573  011216  000137  011502                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
4574  011222                          10$:
4575  011222  004737  043762                  JSR     PC,GETSTS
4576  011226  004737  044046                  JSR     PC,GET              ;GO READ REGISTERS VIA GET SUB
4577  011232  000403                          BR      20$     ;GO TO 20$ IF NO ERROR
4578  011234  000240                          NOP                         ;RETURN HERE IF ERROR
4579  011236  104000                          ERROR                       ;ERROR DEFINED BY GET SUB
4580  011240  000520                          BR      100$                ;SKIP REST OF TEST
4581  011242  032737  010000  001340  20$:    BIT     #MOL,RMDSI          ;MEDIUM ON LINE??
4582  011250  001016                          BNE     30$                 ;YES!!
4583  011252  013737  001340  001140          MOV     RMDSI,$GDDAT        ;EXPECTED STATUS
4584  011260  042737  162000  001140          BIC     #ATA!ERR!PIP!LST,$GDDAT
4585  011266  052737  010000  001140          BIS     #MOL,$GDDAT
4586  011274  013737  001340  001142          MOV     RMDSI,$BDDAT        ;RECEIVED STATUS
4587  011302  104151                          ERROR   151                 ;MOL STATUS INCORRECT
4588  011304  000462                          BR      80$
4589
4590  011306  032737  000200  001370  30$:    BIT     #DVC,RMER2I         ;IS THERE A DEVICE CHECK??
4591  011314  001406                          BEQ     50$
4592  011316  013737  001370  001142          MOV     RMER2I,$BDDAT       ;RECEIVED STATUS
4593  011324  005037  001140                  CLR     $GDDAT              ;EXPECTED STATUS
4594  011330  104152                          ERROR   152                 ;DRIVE FAULT
4595
4596  011332  032737  000200  001370  50$:    BIT     #DVC,RMER2I         ;WAS DVC SET??
4597  011340  001414                          BEQ     60$                 ;NO!!
4598  011342  032737  040000  001342          BIT     #UNS,RMER1I         ;IS UNS SET??
4599  011350  001022                          BNE     70$                 ;YES!!
4600  011352  013737  001342  001142          MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
4601  011360  012737  040000  001140          MOV     #UNS,$GDDAT         ;EXPECTED STATUS
4602  011366  104153                          ERROR   153                 ;DVC IS SET BUT UNS IS NOT
4603  011370  000412                          BR      70$
4604  011372  032737  040000  001342  60$:    BIT     #UNS,RMER1I         ;IS UNS SET??
4605  011400  001406                          BEQ     70$                 ;NO!!
4606  011402  013737  001342  001142          MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
4607  011410  005037  001140                  CLR     $GDDAT              ;EXPECTED STATUS
4608  011414  104154                          ERROR   154                 ;DVC IS NOT SET BUT UNS IS
4609  011416  032737  004000  001340  70$:    BIT     #WRL,RMDSI          ;IS WRITE PROTECT ON??
4610  011424  001414                          BEQ     90$                 ;NO!!
4611  011426  013737  001340  001140          MOV     RMDSI,$GDDAT        ;EXPECTED STATUS
4612  011434  042737  166076  001140          BIC     #^C<MOL!PGM!DPR!DRY!VV!OM>,$GDDAT
4613  011442  013737  001340  001142          MOV     RMDSI,$BDDAT        ;RECEIVED STATUS
4614  011450  104175                          ERROR   175                 ;SELECTED DEVICE IN WRITE PROTECT
4615  011452  000137  042000          80$:    JMP     $EOSP   ;SKIP REST OF TEST
4616
4617  011456  032737  040000  001370  90$:    BIT     #SKI,RMER2I         ;IS SKI SET??
4618  011464  001406                          BEQ     100$                ;NO!!
4619  011466  013737  001370  001142          MOV     RMER2I,$BDDAT       ;RECEIVED STATUS
4620  011474  005037  001140                  CLR     $GDDAT              ;EXPECTED STATUS
4621  011500  104205                          ERROR   205                 ;PERSISTENT SEEK INCOMPLETE ERROR
4622  011502                          100$:
4623
4624                                  ;;*****************************************************************
4625                                  ;*TEST 10       PRIMARY/SECONDARY ERROR TEST
4626                                  ;;*****************************************************************
```

```
4627  011502                                  TST10:
4628  011502  000004                                  SCOPE                   ;SCOPE CALL
4629  011504  000240                                  NOP                     ;START OF TEST
4630  011506  012706  001100                         MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4631  011512  013700  001276                         MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4632  011516  013701  001446                         MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4633  011522  012737  000010  001226                 MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4634
4635  011530  004737  053210                         JSR     PC,CNTCLR
4636  011534  000404                                  BR      1$      ;GO TO 1$ IF NO ERROR
4637  011536  000240                                  NOP                     ;RETURN HERE IF ERROR
4638  011540  104000                                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
4639  011542  000137  011662                         JMP     5$      ;GO TO 5$ IF ERROR WAS FOUND
4640  011546                                  1$:
4641  011546  112737  000000  001533                 MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4642  011554  112737  000200  001534                 MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4643  011562  012737  000023  001376                 MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
4644  011570  004737  044316                         JSR     PC,PUT                  ;CALL PUT SUBROUTINE
4645  011574  000403                                  BR      2$      ;GO TO 2$ IF NO ERROR
4646  011576  000240                                  NOP                     ;RETURN HERE TO REPORT AN ERROR
4647  011600  104000                                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4648  011602  000427                                  BR      5$              ;SKIP REST OF TEST IF ERROR
4649  011604  004737  043762          2$:             JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
4650  011610  004737  044656                         JSR     PC,TIMOUT       ;WAIT FOR COMPLETION OF NOP
4651  011614  004737  044046                         JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4652  011620  000403                                  BR      3$      ;GO TO 3$ IF NO ERROR
4653  011622  000240                                  NOP                     ;RETURN HERE IF ERROR
4654  011624  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
4655  011626  000415                                  BR      5$      ;SKIP REST OF TEST IF ERROR
4656  011630                                  3$:
4657  011630  004737  045042                         JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4658  011634  000404                                  BR      4$              ;GO TO 4$ IF NO ERROR
4659  011636  000240                                  NOP                     ;RETURN HERE IF ERROR
4660  011640  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4661  011642  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4662  011644  000406                                  BR      5$              ;SKIP REST OF TEST IF ERROR
4663  011646                                  4$:
4664  011646  004737  045674                         JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4665  011652  000403                                  BR      5$              ;GO TO 5$ IF NO ERROR
4666  011654  000240                                  NOP                     ;RETURN HERE IF ERROR
4667  011656  104000                                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4668  011660  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4669  011662                                  5$:
4670
4671                                          ;;*****************************************************************
4672                                          ;*TEST 11        DIAGNOSTIC MODE TEST
4673                                          ;;*****************************************************************
4674  011662                                  TST11:
4675  011662  000004                                  SCOPE                   ;SCOPE CALL
4676  011664  000240                                  NOP                     ;START OF TEST
4677  011666  012706  001100                         MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4678  011672  013700  001276                         MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4679  011676  013701  001446                         MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4680  011702  012737  000011  001226                 MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4681
4682  011710  004737  053210                         JSR     PC,CNTCLR
```

```
 4683  011714  000404                          BR      1$      ;GO TO 1$ IF NO ERROR
 4684  011716  000240                          NOP                     ;RETURN HERE IF ERROR
 4685  011720  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
 4686  011722  000137  012734                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
 4687  011726                          1$:
 4688  011726  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
 4689  011734  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
 4690  011742  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
 4691  011750  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
 4692  011754  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
 4693  011756  000240                          NOP                     ;RETURN HERE IF ERROR
 4694  011760  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
 4695  011762  000137  012734                  JMP     22$             ;GO TO 22$ IF ERROR WAS FOUND
 4696  011766  004737  043762          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
 4697  011772  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4698  011776  000404                          BR      3$      ;GO TO 3$ IF NO ERROR
 4699  012000  000240                          NOP                     ;RETURN HERE IF ERROR
 4700  012002  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 4701  012004  000137  012734                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
 4702  012010  032737  000001  001352  3$:     BIT     #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
 4703  012016  001011                          BNE     5$              ;YES!!
 4704  012020  012737  000001  001140          MOV     #DMD,$GDDAT     ;EXPECTED STATUS
 4705  012026  013737  001352  001142          MOV     RMMR1I,$BDDAT   ;RECEIVED STATUS
 4706  012034  104176                          ERROR   176             ;COULD NOT SET DIAGNOSTIC MODE
 4707  012036  000137  012734                  JMP     22$             ;SKIP REST OF TEST IF DMD = 0
 4708  012042  032737  010000  001340  5$:     BIT     #MOL,RMDSI      ;IS "MOL" = 0   ??
 4709  012050  001411                          BEQ     6$              ;YES!!
 4710  012052  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4711  012060  042737  167777  001142          BIC     #^CMOL,$BDDAT
 4712  012066  005037  001140                  CLR     $GDDAT          ;SETUP GOOD DATA FOR TYPEOUT
 4713  012072  104177                          ERROR   177             ;INCORRECT MOL STATUS
 4714  012074  032737  020000  001340  6$:     BIT     #PIP,RMDSI      ;IS PIP  SET??
 4715  012102  001012                          BNE     7$              ;YES!!
 4716  012104  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4717  012112  042737  157777  001142          BIC     #^CPIP,$BDDAT
 4718  012120  012737  020000  001140          MOV     #PIP,$GDDAT     ;EXPECTED PIP SET
 4719  012126  104200                          ERROR   200             ;INCORRECT PIP STATUS
 4720  012130  032737  004000  001340  7$:     BIT     #WRL,RMDSI      ;IS WRITE LOCK OFF??
 4721  012136  001411                          BEQ     8$              ;YES!!
 4722  012140  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4723  012146  042737  173777  001142          BIC     #^CWRL,$BDDAT
 4724  012154  005037  001140                  CLR     $GDDAT          ;EXPECTED WRL  = 0
 4725  012160  104201                          ERROR   201             ;INCORRECT WRL STATUS
 4726  012162  032737  040000  001370  8$:     BIT     #SKI,RMER2I     ;IS SKI = 0
 4727  012170  001411                          BEQ     9$              ;YES!!
 4728  012172  013737  001370  001142          MOV     RMER2I,$BDDAT   ;SETUP BAD DATA FOR TYPEOUT
 4729  012200  042737  137777  001142          BIC     #^CSKI,$BDDAT
 4730  012206  005037  001140                  CLR     $GDDAT          ;SKI SHOULD BE 0
 4731  012212  104202                          ERROR   202             ;REPORT BAD SKI STATUS
 4732  012214  032737  000200  001370  9$:     BIT     #DVC,RMER2I     ;IS DEVICE CHECK  = 0??
 4733  012222  001411                          BEQ     10$             ;YES!!
 4734  012224  013737  001370  001142          MOV     RMER2I,$BDDAT   ;SETUP BAD DATA FOR  TYPEOUT
 4735  012232  042737  177577  001142          BIC     #^CDVC,$BDDAT
 4736  012240  005037  001140                  CLR     $GDDAT          ;DVC SHOULD BE 0
 4737  012244  104203                          ERROR   203             ;REPORT BAD DVC STATUS
 4738  012246                          10$:
```

```
4739  012246  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4740  012254  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4741  012262  012737  001711  001422          MOV     #DMD!MUR!MOC!MSER!MDF!MWP,R0    ;RMMR1=DMD!MUR!MOC!MSER!MDF!MWP
4742  012270  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4743  012274  000404                          BR      11$             ;GO TO 11$ IF NO ERROR
4744  012276  000240                          NOP                     ;RETURN HERE IF ERROR
4745  012300  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4746  012302  000137  012734                  JMP     22$             ;GO TO 22$ IF ERROR WAS FOUND
4747  012306                          11$:
4748  012306  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4749  012312  000404                          BR      12$     ;GO TO 12$ IF NO ERROR
4750  012314  000240                          NOP                     ;RETURN HERE IF ERROR
4751  012316  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4752  012320  000137  012734                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4753  012324  032737  000001  001352  12$:    BIT     #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
4754  012332  001011                          BNE     125$            ;YES!!
4755  012334  012737  000001  001140          MOV     #DMD,$GDDAT     ;EXPECTED STATUS
4756  012342  013737  001352  001142          MOV     RMMR1I,$BDDAT   ;RECEIVED STATUS
4757  012350  104176                          ERROR   176             ;COULD NOT SET DIAGNOSTIC MODE
4758  012352  000137  012734                  JMP     22$             ;SKIP REST OF TEST IF DMD = 0
4759  012356  032737  010000  001340  125$:   BIT     #MOL,RMDSI      ;IS MOL = 1??
4760  012364  001012                          BNE     13$             ;YES!!
4761  012366  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4762  012374  042737  167777  001142          BIC     #^CMOL,$BDDAT
4763  012402  012737  010000  001140          MOV     #MOL,$GDDAT     ;EXPECTED MOL  =1
4764  012410  104177                          ERROR   177             ;REPORT MOL STATUS ERROR
4765  012412  032737  020000  001340  13$:    BIT     #PIP,RMDSI      ;IS   PIP 0 ??
4766  012420  001411                          BEQ     14$             ;YES!!
4767  012422  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4768  012430  042737  157777  001142          BIC     #^CPIP,$BDDAT
4769  012436  005037  001140                  CLR     $GDDAT          ;EXPECTED PIP STATUS
4770  012442  104200                          ERROR   200             ;REPORT BAD PIP STATUS
4771  012444  032737  004000  001340  14$:    BIT     #WRL,RMDSI      ;IS WRL SET??
4772  012452  001012                          BNE     15$             ;YES!!
4773  012454  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4774  012462  042737  173777  001142          BIC     #^CWRL,$BDDAT
4775  012470  012737  004000  001140          MOV     #WRL,$GDDAT     ;EXPECTED GOOD STATUS
4776  012476  104201                          ERROR   201             ;REPORT ERROR IN  WRL STATUS
4777  012500  032737  040000  001370  15$:    BIT     #SKI,RMER2I     ;IS SKI SET??
4778  012506  001012                          BNE     16$             ;YES!!
4779  012510  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
4780  012516  042737  137777  001142          BIC     #^CSKI,$BDDAT
4781  012524  012737  040000  001140          MOV     #SKI,$GDDAT     ;EXPECTED SKI  ON
4782  012532  104202                          ERROR   202
4783  012534  032737  000200  001370  16$:    BIT     #DVC,RMER2I     ;IS DVC SET  ??
4784  012542  001012                          BNE     17$             ;YES!!
4785  012544  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
4786  012552  042737  177577  001142          BIC     #^CDVC,$BDDAT
4787  012560  012737  000200  001140          MOV     #DVC,$GDDAT     ;EXPECTED DVC ON
4788  012566  104203                          ERROR   203             ;REPORT DVC STATUS ERROR
4789  012570  032737  000100  001340  17$:    BIT     #VV,RMDSI       ;MUR SHOULD HAVE RESET VOLUME VALID
4790  012576  001411                          BEQ     19$             ;BRANCH IF IT DID
4791  012600  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
4792  012606  042737  177677  001142          BIC     #^CVV,$BDDAT
4793  012614  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
4794  012620  104204                          ERROR   204
```

```
4795  012622
4796  012622  112737  000000  001533  19$:    MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4797  012630  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4798  012636  012737  000011  001376          MOV     #DRVCLR!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = DRVCLR!GO
4799  012644  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
4800  012650  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
4801  012652  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
4802  012654  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4803  012656  000137  012734                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4804  012662                          20$:
4805  012662  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4806  012666  000404                          BR      21$     ;GO TO 21$ IF NO ERROR
4807  012670  000240                          NOP                     ;RETURN HERE IF ERROR
4808  012672  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4809  012674  000137  012734                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4810  012700  032737  002000  001370  21$:    BIT #LBC,RMER2I         ;IS LBC SET ??
4811  012706  001012                          BNE 22$                 ;YES!!
4812  012710  013737  001370  001142          MOV RMER2I,$BDDAT       ;BAD DATA FOR TYPEOUT
4813  012716  012737  002000  001140          MOV #LBC, $GDDAT        ;GOOD DATA FOR TYPEOUT
4814  012724  042737  002000  001142          BIC #LBC, $BDDAT
4815  012732  104262                          ERROR   262
4816  012734                          22$:
4817                                  ;;*****************************************************************
4818                                  ;*TEST 12       PACK ACKNOWLEDGE TEST
4819
4820                                  ;;*****************************************************************
4821  012734                          TST12:
4822  012734  000004                          SCOPE                   ;SCOPE CALL
4823  012736  000240                          NOP                     ;START OF TEST
4824  012740  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4825  012744  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4826  012750  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4827  012754  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4828
4829  012762  004737  053210                  JSR     PC,CNTCLR
4830  012766  000404                          BR      1$      ;GO TO 1$ IF NO ERROR
4831  012770  000240                          NOP                     ;RETURN HERE IF ERROR
4832  012772  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4833  012774  000137  013234                  JMP     9$      ;GO TO 9$ IF ERROR WAS FOUND
4834  013000                          1$:
4835  013000  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4836  013006  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4837  013014  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
4838  013022  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4839  013026  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
4840  013030  000240                          NOP                     ;RETURN HERE IF ERROR
4841  013032  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4842  013034  000137  013234                  JMP     9$              ;GO TO 9$ IF ERROR WAS FOUND
4843  013040                          2$:
4844  013040  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4845  013046  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4846  013054  012737  000000  001422          MOV     #0,RMMR10       ;RMMR1=0
4847  013062  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4848  013066  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
4849  013070  000240                          NOP                     ;RETURN HERE IF ERROR
4850  013072  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
```

```
4851  013074  000137  013234            JMP     9$              ;GO TO 9$ IF ERROR WAS FOUND
4852  013100                          3$:
4853  013100  112737  000000  001533    MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4854  013106  112737  000200  001534    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4855  013114  012737  000023  001376    MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
4856  013122  004737  044316            JSR     PC,PUT          ;CALL PUT SUBROUTINE
4857  013126  000404                    BR      4$      ;GO TO 4$ IF NO ERROR
4858  013130  000240                    NOP                     ;RETURN HERE TO REPORT AN ERROR
4859  013132  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4860  013134  000137  013234            JMP     9$      ;GO TO 9$ IF ERROR WAS FOUND
4861  013140  004737  043762          4$:       JSR     PC,GETST$
4862  013144  004737  044656            JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
4863  013150  004737  044046            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4864  013154  000403                    BR      5$      ;GO TO 5$ IF NO ERROR
4865  013156  000240                    NOP                     ;RETURN HERE IF ERROR
4866  013160  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
4867  013162  000424                    BR      9$              ;SKIP REMAINDER OF TEST
4868  013164                          5$:
4869  013164  004737  045042            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4870  013170  000404                    BR      6$                      ;GO TO 6$ IF NO ERROR
4871  013172  000240                    NOP                     ;RETURN HERE IF ERROR
4872  013174  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4873  013176  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4874  013200  000415                    BR      9$              ;SKIP TO NEXT TEST
4875  013202                          6$:
4876  013202  004737  054206            JSR     PC,ACKSTS       ;GO VERIFY PACK ACKNOWLEDGE
4877  013206  000404                    BR      7$                      ;GO TO 7$ IF NO ERROR
4878  013210  000240                    NOP                     ;RETURN HERE IF ERROR
4879  013212  104000                    ERROR                   ;ERROR # DEFINED BY ACKSTS SUBROUTINE
4880  013214  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4881  013216  000406                    BR      9$              ;SKIP TO NEXT TEST
4882
4883  013220                          7$:
4884  013220  004737  045674            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4885  013224  000403                    BR      9$                      ;GO TO 9$ IF NO ERROR
4886  013226  000240                    NOP                     ;RETURN HERE IF ERROR
4887  013230  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4888  013232  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4889  013234                          9$:
4890
4891                                  ;;*****************************************************************
4892                                  ;*TEST 13        RECALIBRATE TEST
4893
4894                                  ;;*****************************************************************
4895  013234                          TST13:
4896  013234  000004                    SCOPE                   ;SCOPE CALL
4897  013236  000240                    NOP                     ;START OF TEST
4898  013240  012737  000001  001206    MOV     #1,STIMES       ;LOAD ITERATION COUNT
4899  013246  012737  013262  001122    MOV     #1$,$LPADR
4900  013254  012737  013262  001124    MOV     #1$,$LPERR
4901  013262                          1$:
4902  013262  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4903  013266  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4904  013272  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4905  013276  012737  000013  001226    MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4906  013304  004737  043046            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
4907  013310  050020                          .WORD   050020  ;TASK DESCRIPTOR
4908  013312  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
4909  013314  000240                          NOP                     ;RETURN HERE IF ERROR
4910  013316  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4911  013320  000137  013466                  JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4912  013324                          5$:
4913  013324  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4914  013332  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4915  013340  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
4916  013346  004737  044316                  JSR     PC,PUT                  ;CALL PUT SUBROUTINE
4917  013352  000404                          BR      6$      ;GO TO 6$ IF NO ERROR
4918  013354  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
4919  013356  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4920  013360  000137  013466                  JMP     10$     ;GO TO 10$ IF ERROR WAS FOUND
4921  013364  004737  043762          6$:     JSR     PC,GETSTS       ;GO SETUP FOR STATUS FETCH
4922  013370  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
4923  013374  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4924  013400  000404                          BR      7$      ;GO TO 7$ IF NO ERROR
4925  013402  000240                          NOP                     ;RETURN HERE IF ERROR
4926  013404  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4927  013406  000137  013466                  JMP     10$     ;GO TO 10$ IF ERROR WAS FOUND
4928  013412                          7$:
4929  013412  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4930  013416  000405                          BR      8$                      ;GO TO 8$ IF NO ERROR
4931  013420  000240                          NOP                     ;RETURN HERE IF ERROR
4932  013422  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4933  013424  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4934  013426  000137  013466                  JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4935  013432                          8$:
4936  013432  004737  055002                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
4937  013436  000405                          BR      9$                      ;GO TO 9$ IF NO ERROR
4938  013440  000240                          NOP                     ;RETURN HERE IF ERROR
4939  013442  104000                          ERROR                   ;ERROR # DEFINED BY RCLSTS SUBROUTINE
4940  013444  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4941  013446  000137  013466                  JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4942  013452                          9$:
4943  013452  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4944  013456  000403                          BR      10$                     ;GO TO 10$ IF NO ERROR
4945  013460  000240                          NOP                     ;RETURN HERE IF ERROR
4946  013462  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4947  013464  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4948  013466                          10$:
4949
4950                                  ;;*****************************************************************
4951                                  ;*TEST 14        ABORT RECALIBRATE TEST
4952                                  ;;*****************************************************************
4953  013466                          TST14:
4954  013466  000004                          SCOPE                   ;SCOPE CALL
4955  013470  000240                          NOP                     ;START OF TEST
4956  013472  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
4957  013500  012737  013514  001122          MOV     #1$,$LPADR
4958  013506  012737  013514  001124          MOV     #1$,$LPERR
4959  013514                          1$:
4960  013514  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4961  013520  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4962  013524  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
4963  013530  012737  000014  001226          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4964
4965  013536  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4966  013542  054130                          .WORD   054130  ;TASK DESCRIPTOR
4967  013544  000404                          BR      8$                      ;GO TO 8$ IF NO ERROR
4968  013546  000240                          NOP                     ;RETURN HERE IF ERROR
4969  013550  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4970  013552  000137  014010                  JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
4971  013556                          8$:
4972  013556  112737  000014  001533          MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
4973  013564  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4974  013572  012737  040000  001412          MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
4975  013600  004737  044316                  JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
4976  013604  000404                          BR      9$      ;GO TO 9$ IF NO ERROR
4977  013606  000240                          NOP                     ;RETURN HERE TO REPORT ERROR
4978  013610  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4979  013612  000137  014010                  JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
4980  013616                          9$:
4931  013616  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4982  013624  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4983  013632  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
4934  013640  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
4935  013644  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4936  013646  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
4937  013650  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4938  013652  000137  014010                  JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
4939  013656  004737  043762          10$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
4990  013662  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4991  013666  000404                          BR      11$     ;GO TO 11$ IF NO ERROR
4992  013670  000240                          NOP                     ;RETURN HERE IF ERROR
4993  013672  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4994  013674  000137  014010                  JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
4995  013700  032737  020000  001340  11$:     BIT     #PIP,RMDSI      ;DID THE DRIVE RECALIBRATE??
4996  013706  001401                          BEQ     12$             ;NO!!
4997  013710  104214                          ERROR   214             ;ERROR SHOULD PREVENT RECALIBRATE
4998  013712  004737  044656          12$:    JSR     PC,TIMOUT       ;WAIT FOR  COMPLETION
4999  013716  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5000  013722  000404                          BR      13$     ;GO TO 13$ IF NO ERROR
5001  013724  000240                          NOP                     ;RETURN HERE IF ERROR
5002  013726  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5003  013730  000137  014010                  JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
5004  013734                          13$:
5005  013734  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5006  013740  000405                          BR      14$                     ;GO TO 14$ IF NO ERROR
5007  013742  000240                          NOP                     ;RETURN HERE IF ERROR
5008  013744  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5009  013746  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5010  013750  000137  014010                  JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5011  013754                          14$:
5012  013754  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5013  013760  000405                          BR      141$                    ;GO TO 141$ IF NO ERROR
5014  013762  000240                          NOP                     ;RETURN HERE IF ERROR
5015  013764  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5016  013766  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5017  013770  000137  014010                  JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5018  013774                          141$:
```

```
5019  013774  004737  045674          T14     ABORT RECALIBRATE TEST
5020  014000  000403                          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5021  014002  000240                          BR      15$                     ;GO TO 15$ IF NO ERROR
5022  014004  104000                          NOP                     ;RETURN HERE IF ERROR
5023  014006  004736                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5024  014010                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5025                                  15$:
5026                                  ;;*****************************************************************
5027                                  ;*TEST 15       IVC RECALIBRATE TEST
5028                                  ;;*****************************************************************
5029  014010                          TST15:
5030  014010  000004                          SCOPE                   ;SCOPE CALL
5031  014012  000240                          NOP                     ;START OF TEST
5032  014014  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5033  014022  012737  014036  001122          MOV     #1$,$LPADR
5034  014030  012737  014036  001124          MOV     #1$,$LPERR
5035  014036                          1$:
5036  014036  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5037  014042  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5038  014046  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5039  014052  012737  000015  001226          MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5040  014060  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5041  014064  054130                          .WORD   054130  ;TASK DESCRIPTOR
5042  014066  000404                          BR      8$                      ;GO TO 8$ IF NO ERROR
5043  014070  000240                          NOP                     ;RETURN HERE IF ERROR
5044  014072  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5045  014074  000137  014334                  JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5046  014100                          8$:
5047  014100  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
5048  014106  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5049  014114  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
5050  014122  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
5051  014126  000404                          BR      9$              ;GO TO 9$ IF NO ERROR
5052  014130  000240                          NOP                     ;RETURN HERE IF ERROR
5053  014132  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5054  014134  000137  014334                  JMP     15$             ;GO TO 15$ IF ERROR WAS FOUND
5055  014140                          9$:
5056  014140  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
5057  014146  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5058  014154  012737  000000  001422          MOV     #0,RMMR10       ;RMMR1=0
5059  014162  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
5060  014166  000404                          BR      10$             ;GO TO 10$ IF NO ERROR
5061  014170  000240                          NOP                     ;RETURN HERE IF ERROR
5062  014172  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5063  014174  000137  014334                  JMP     15$             ;GO TO 15$ IF ERROR WAS FOUND
5064  014200                          10$:
5065  014200  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5066  014206  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5067  014214  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
5068  014222  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5069  014226  000404                          BR      11$     ;GO TO 11$ IF NO ERROR
5070  014230  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5071  014232  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5072  014234  000137  014334                  JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
5073  014240  004737  043762          11$:    JSR     PC,GETST$       ;SETUP FOR STATUS FETCH
5074  014244  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
```

```
5075  014250  004737  044046          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5076  014254  000404                  BR      12$     ;GO TO 12$ IF NO ERROR
5077  014256  000240                  NOP                     ;RETURN HERE IF ERROR
5078  014260  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
5079  014262  000137  014334          JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
5080  014266                  12$:
5081  014266  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5082  014272  000405                  BR      13$                     ;GO TO 13$ IF NO ERROR
5083  014274  000240                  NOP                     ;RETURN HERE IF ERROR
5084  014276  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5085  014300  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5086  014302  000137  014334          JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5087  014306  032737  010000  001370  13$:    BIT     #IVC,RMLR2I     ;IVC SHOULD BE SET
5088  014314  001001                  BNE     14$                     ;IT IS - GO CHECK SECONDARY ERRORS
5089  014316  104215                  ERROR   215             ;IVC NOT SET DURING RECALIBRATE
5090  014320                  14$:
5091  014320  004737  045674          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5092  014324  000403                  BR      15$                     ;GO TO 15$ IF NO ERROR
5093  014326  000240                  NOP                     ;RETURN HERE IF ERROR
5094  014330  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5095  014332  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5096  014334                  15$:
5097
5098                          ;;*****************************************************************
5099                          ;*TEST 16       IAE RECALIBRATE TEST
5100                          ;;*****************************************************************
5101  014334                  TST16:
5102  014334  000004                  SCOPE                   ;SCOPE CALL
5103  014336  000240                  NOP                     ;START OF TEST
5104  014340  012737  000001  001206  MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5105  014346  012737  014362  001122  MOV     #1$,$LPADR
5106  014354  012737  014362  001124  MOV     #1$,$LPERR
5107  014362                  1$:
5108  014362  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5109  014366  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5110  014372  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5111  014376  012737  000016  001226  MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5112  014404  004737  043046          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5113  014410  054130                  .WORD   054130  ;TASK DESCRIPTOR
5114  014412  000404                  BR      70$                     ;GO TO 70$ IF NO ERROR
5115  014414  000240                  NOP                     ;RETURN HERE IF ERROR
5116  014416  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5117  014420  000137  014612          JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
5118  014424                  70$:
5119  014424  012737  177777  001432  MOV     #-1,RMDC0       ;RMDC WILL BE ALL ONES
5120  014432  012737  177777  001404  MOV     #-1,RMDA0       ;RMDA WILL BE ALL ONES
5121  014440  012737  000007  001376  MOV     #RECAL!GO,RMCS10
5122  014446  012702  001533          MOV     #PUTINX,R2      ;R2 POINTS TO PUT INDEX TABLE
5123  014452  112722  000034          MOVB    #RMDC,(R2)+     ;SETUP PUT INDEX TABLE
5124  014456  112722  000006          MOVB    #RMDA,(R2)+
5125  014462  112722  000000          MOVB    #RMCS1,(R2)+
5126  014466  112722  000200          MOVB    #200,(R2)+
5127  014472  004737  044316          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5128  014476  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
5129  014500  000240                  NOP                     ;RETURN HERE IF ERROR
5130  014502  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
```

```
5131  014504  000137  014612                JMP     120$    ;GO TO 120$ IF ERROR WAS FOUND
5132  014510  004737  043762        80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5133  014514  004737  044656                JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
5134  014520  004737  044046                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5135  014524  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5136  014526  000240                        NOP                     ;RETURN HERE IF ERROR
5137  014530  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5138  014532  000137  014612                JMP     120$    ;GO TO 120$ IF ERROR WAS FOUND
5139  014536                        90$:
5140  014536  004737  045042                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5141  014542  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5142  014544  000240                        NOP                     ;RETURN HERE IF ERROR
5143  014546  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5144  014550  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5145  014552  000137  014612                JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
5146  014556                        100$:
5147  014556  004737  055002                JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
5148  014562  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5149  014564  000240                        NOP                     ;RETURN HERE IF ERROR
5150  014566  104000                        ERROR                   ;ERROR # DEFINED BY RCLSTS SUBROUTINE
5151  014570  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5152  014572  000137  014612                JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
5153  014576                        110$:
5154  014576  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5155  014602  000403                        BR      120$                    ;GO TO 120$ IF NO ERROR
5156  014604  000240                        NOP                     ;RETURN HERE IF ERROR
5157  014606  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5158  014610  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5159  014612                        120$:
5160
5161                                ;;*****************************************************************
5162                                ;*TEST 17       RECALIBRATE AT OFFSET
5163                                ;;*****************************************************************
5164  014612                        TST17:
5165  014612  000004                        SCOPE                   ;SCOPE CALL
5166  014614  000240                        NOP                     ;START OF TEST
5167  014616  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5168  014624  012737  014640  001122        MOV     #1$,$LPADR
5169  014632  012737  014640  001124        MOV     #1$,$LPERR
5170  014640                        1$:
5171  014640  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5172  014644  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5173  014650  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5174  014654  012737  000017  001226        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5175
5176  014662  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5177  014666  050020                        .WORD   050020  ;TASK DESCRIPTOR
5178  014670  000404                        BR      5$                      ;GO TO 5$ IF NO ERROR
5179  014672  000240                        NOP                     ;RETURN HERE IF ERROR
5180  014674  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5181  014676  000137  015104                JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
5182  014702                        5$:
5183  014702  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5184  014710  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5185  014716  012737  000015  001376        MOV     #OFFSET!GO,RMCS1        ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5186  014724  004737  044316                JSR     PC,PUT          ;CALL PUT SUBROUTINE
```

```
5187  014730  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
5188  014732  000240                          NOP             ;RETURN HERE TO REPORT AN ERROR
5189  014734  104000                          ERROR           ;ERROR NUMBER DEFINED BY PUT SUB
5190  014736  000137  015104                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
5191  014742                          10$:
5192  014742  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5193  014750  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5194  014756  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
5195  014764  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5196  014770  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
5197  014772  000240                          NOP             ;RETURN HERE TO REPORT AN ERROR
5198  014774  104000                          ERROR           ;ERROR NUMBER DEFINED BY PUT SUB
5199  014776  000137  015104                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
5200  015002                          20$:
5201  015002  004737  043762                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5202  015006                          30$:
5203  015006  004737  044656                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5204  015012                          40$:
5205  015012  004737  044046                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5206  015016  000404                          BR      50$             ;GO TO 50$ IF NO ERROR
5207  015020  000240                          NOP             ;RETURN HERE IF ERROR
5208  015022  104000                          ERROR           ;ERROR # DEFINED BY GET SUBROUTINE
5209  015024  000137  015104                  JMP     80$             ;GO TO 80$ IF ERROR WAS FOUND
5210  015030                          50$:
5211  015030  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5212  015034  000405                          BR      60$             ;GO TO 60$ IF NO ERROR
5213  015036  000240                          NOP             ;RETURN HERE IF ERROR
5214  015040  104000                          ERROR           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5215  015042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5216  015044  000137  015104                  JMP     80$             ;GO TO 80$ IF ERROR WAS FOUND
5217  015050                          60$:
5218  015050  004737  055002                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
5219  015054  000405                          BR      70$             ;GO TO 70$ IF NO ERROR
5220  015056  000240                          NOP             ;RETURN HERE IF ERROR
5221  015060  104000                          ERROR           ;ERROR # DEFINED BY RCLSTS SUBROUTINE
5222  015062  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5223  015064  000137  015104                  JMP     80$             ;GO TO 80$ IF ERROR WAS FOUND
5224  015070                          70$:
5225  015070  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5226  015074  000403                          BR      80$             ;GO TO 80$ IF NO ERROR
5227  015076  000240                          NOP             ;RETURN HERE IF ERROR
5228  015100  104000                          ERROR           ;ERROR # DEFINED BY SECERR SUBROUTINE
5229  015102  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5230  015104                          80$:
5231                                  ;;*****************************************************************
5232                                  ;*TEST 20        DRIVE CLEAR TEST
5233
5234                                  ;;*****************************************************************
5235  015104                          TST20:
5236  015104  000004                          SCOPE                   ;SCOPE CALL
5237  015106  000240                          NOP                     ;START OF TEST
5238  015110  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5239  015114  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5240  015120  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5241  015124  012737  000020  001226          MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5242  015132  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
015136  054130                          .WORD   054130  ;TASK DESCRIPTOR
015140  000404                          BR      2$                      ;GO TO 2$ IF NO ERROR
015142  000240                          NOP                     ;RETURN HERE IF ERROR
015144  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015146  000137  015354                  JMP     21$                     ;GO TO 21$ IF ERROR WAS FOUND
015152                          2$:
015152  012737  000000  001404          MOV     #0,RMDA0
015160  012737  000011  001376          MOV     #DRVCLR!GO,RMCS10
015166  012737  177777  001412          MOV     #-1,RMER10
015174  012737  177777  001440          MOV     #-1,RMER20
015202  042737  004000  001440          BIC     #LSC,RMER20     ;DELETE FOR PASS 2 ETCH
015210  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
015214  112722  000006                  MOVB    #RMDA,(R2)+     ;RMDA CLEARS LBT
015220  112722  000042                  MOVB    #RMER2,(R2)+
015224  112722  000014                  MOVB    #RMER1,(R2)+
015230  112722  000000                  MOVB    #RMCS1,(R2)+
015234  112722  000200                  MOVB    #200,(R2)+
015240  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
015244  000403                          BR      3$      ;GO TO 3$ IF NO ERROR
015246  000240                          NOP                     ;RETURN HERE IF ERROR
015250  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
015252  000440                          BR      21$             ;ESCAPE IF ERROR
015254  004737  043762          3$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
015260  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR DRIVE CLEAR TO COMPLETE
015264  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
015270  000403                          BR      4$      ;GO TO 4$ IF NO ERROR
015272  000240                          NOP                     ;RETURN HERE IF ERROR
015274  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
015276  000426                          BR      21$             ;SKIP REMAINDER OF TEST
015300                          4$:
015300  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
015304  000405                          BR      5$                      ;GO TO 5$ IF NO ERROR
015306  000240                          NOP                     ;RETURN HERE IF ERROR
015310  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
015312  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015314  000137  015354                  JMP     21$                     ;GO TO 21$ IF ERROR WAS FOUND
015320                          5$:
015320  004737  056544                  JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR
015324  000405                          BR      6$                      ;GO TO 6$ IF NO ERROR
015326  000240                          NOP                     ;RETURN HERE IF ERROR
015330  104000                          ERROR                   ;ERROR # DEFINED BY DRVSTS SUBROUTINE
015332  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015334  000137  015354                  JMP     21$                     ;GO TO 21$ IF ERROR WAS FOUND
015340                          6$:
015340  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
015344  000403                          BR      21$                     ;GO TO 21$ IF NO ERROR
015346  000240                          NOP                     ;RETURN HERE IF ERROR
015350  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
015352  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015354                          21$:

                                ;;*****************************************************
                                ;*TEST 21       NOP TEST

                                ;;*****************************************************
015354                          TST21:
```

```
5299  015354  000004                          SCOPE                   ;SCOPE CALL
5300  015356  000240                          NOP                     ;START OF TEST
5301  015360  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5302  015364  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5303  015370  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5304  015374  012737  000021  001226          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5305  015402  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5306  015406  054130                          .WORD   054130  ;TASK DESCRIPTOR
5307  015410  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5308  015412  000240                          NOP                     ;RETURN HERE IF ERROR
5309  015414  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5310  015416  000137  015604                  JMP     130$                    ;GO TO 130$ IF ERROR WAS FOUND
5311  015422                          70$:
5312  015422  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5313  015430  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5314  015436  012737  000001  001376          MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
5315  015444  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5316  015450  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5317  015452  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5318  015454  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5319  015456  000137  015604                  JMP     130$    ;GO TO 130$ IF ERROR WAS FOUND
5320  015462  004737  043762          80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5321  015466  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5322  015472  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5323  015474  000240                          NOP                     ;RETURN HERE IF ERROR
5324  015476  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5325  015500  000137  015604                  JMP     130$    ;GO TO 130$ IF ERROR WAS FOUND
5326  015504                          90$:
5327  015504  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5328  015510  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5329  015512  000240                          NOP                     ;RETURN HERE IF ERROR
5330  015514  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5331  015516  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5332  015520  000137  015604                  JMP     130$                    ;GO TO 130$ IF ERROR WAS FOUND
5333  015524                          100$:
5334  015524  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5335  015530  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
5336  015532  000010                          .WORD   DPE             ;MASK FOR RMER2
5337  015534  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5338  015536  000240                          NOP                     ;RETURN HERE IF ERROR
5339  015540  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5340  015542  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5341  015544  000137  015604                  JMP     130$                    ;GO TO 130$ IF ERROR WAS FOUND
5342  015550                          110$:
5343  015550  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5344  015554  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
5345  015556  000240                          NOP                     ;RETURN HERE IF ERROR
5346  015560  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5347  015562  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5348  015564  000137  015604                  JMP     130$                    ;GO TO 130$ IF ERROR WAS FOUND
5349  015570                          120$:
5350  015570  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5351  015574  000403                          BR      130$                    ;GO TO 130$ IF NO ERROR
5352  015576  000240                          NOP                     ;RETURN HERE IF ERROR
5353  015600  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5354  015602  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
 5355  015604                                  130$:
 5356
 5357                                          ;;*****************************************************************
 5358                                          ;*TEST 22       OFFSET TEST
 5359
 5360                                          ;;*****************************************************************
 5361  015604                                  TST22:
 5362  015604  000004                                 SCOPE                   ;SCOPE CALL
 5363  015606  000240                                 NOP                     ;START OF TEST
 5364  015610  012706  001100                         MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 5365  015614  013700  001276                         MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 5366  015620  013701  001446                         MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 5367  015624  012737  000022  001226                 MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 5368  015632  004737  043046                         JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 5369  015636  054130                                 .WORD   054130  ;TASK DESCRIPTOR
 5370  015640  000404                                 BR      40$                     ;GO TO 40$ IF NO ERROR
 5371  015642  000240                                 NOP                     ;RETURN HERE IF ERROR
 5372  015644  104000                                 ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 5373  015646  000137  016130                         JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
 5374  015652                                  40$:
 5375  015652  112737  000000  001533                 MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
 5376  015660  112737  000200  001534                 MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
 5377  015666  012737  000015  001376                 MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
 5378  015674  004737  044316                         JSR     PC,PUT          ;CALL PUT SUBROUTINE
 5379  015700  000404                                 BR      50$     ;GO TO 50$ IF NO ERROR
 5380  015702  000240                                 NOP                     ;RETURN HERE TO REPORT AN ERROR
 5381  015704  104000                                 ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
 5382  015706  000137  016130                         JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
 5383  015712  004737  043762          50$:           JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
 5384  015716  004737  044656                         JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
 5385  015722  004737  044046                         JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 5386  015726  000404                                 BR      60$     ;GO TO 60$ IF NO ERROR
 5387  015730  000240                                 NOP                     ;RETURN HERE IF ERROR
 5388  015732  104000                                 ERROR                   ;ERROR DEFINED BY GET SUB
 5389  015734  000137  016130                         JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
 5390  015740                                  60$:
 5391  015740  004737  045042                         JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 5392  015744  000405                                 BR      70$                     ;GO TO 70$ IF NO ERROR
 5393  015746  000240                                 NOP                     ;RETURN HERE IF ERROR
 5394  015750  104000                                 ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 5395  015752  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5396  015754  000137  016130                         JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
 5397  015760  032737  000001  001340  70$:           BIT     #OM,RMDSI       ;OFFSET MODE ON??
 5398  015766  001012                                 BNE     80$             ;YES!!
 5399  015770  013737  001340  001142                 MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
 5400  015776  042737  177776  001142                 BIC     #^COM,$BDDAT
 5401  016004  012737  000001  001140                 MOV     #OM,$GDDAT      ;GOOD DATA FOR TYPEOUT
 5402  016012  104156                                 ERROR   156             ;OFFSET MODE NOT SET
 5403  016014  032737  100000  001340  80$:           BIT     #ATA,RMDSI      ;WAS ATTENTION SET ??
 5404  016022  001012                                 BNE     90$             ;YES!!
 5405  016024  013737  001340  001142                 MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
 5406  016032  042737  077777  001142                 BIC     #^CATA,$BDDAT
 5407  016040  012737  100000  001140                 MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
 5408  016046  104172                                 ERROR   172             ;ATTENTION NOT SET
 5409  016050                                  90$:
 5410  016050  004737  061416                         JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
```

```
5411  016054  115760                        .WORD   NOTMSK          ;MASK FOR RMER1
5412  016056  000010                        .WORD   DPE             ;MASK FOR RMER2
5413  016060  000405                        BR      91$                     ;GO TO 91$ IF NO ERROR
5414  016062  000240                        NOP                     ;RETURN HERE IF ERROR
5415  016064  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5416  016066  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5417  016070  000137  016130                JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
5418  016074                          91$:
5419  016074  004737  060712                JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5420  016100  000405                        BR      92$                     ;GO TO 92$ IF NO ERROR
5421  016102  000240                        NOP                     ;RETURN HERE IF ERROR
5422  016104  104000                        ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5423  016106  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5424  016110  000137  016130                JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
5425  016114                          92$:
5426  016114  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5427  016120  000403                        BR      100$                    ;GO TO 100$ IF NO ERROR
5428  016122  000240                        NOP                     ;RETURN HERE IF ERROR
5429  016124  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5430  016126  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5431  016130                          100$:
5432
5433                                  ;;*****************************************************************
5434                                  ;*TEST 23       GO/ATA TEST
5435                                  ;;*****************************************************************
5436  016130                          TST23:
5437  016130  000004                        SCOPE                   ;SCOPE CALL
5438  016132  000240                        NOP                     ;START OF TEST
5439  016134  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5440  016140  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5441  016144  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5442  016150  012737  000023  001226        MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5443
5444  016156  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5445  016162  054130                        .WORD   054130  ;TASK DESCRIPTOR
5446  016164  000404                        BR      70$                     ;GO TO 70$ IF NO ERROR
5447  016166  000240                        NOP                     ;RETURN HERE IF ERROR
5448  016170  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5449  016172  000137  016550                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5450  016176                          70$:
5451  016176  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5452  016204  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5453  016212  012737  000015  001376        MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5454  016220  004737  044316                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5455  016224  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
5456  016226  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
5457  016230  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5458  016232  000137  016550                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5459  016236  004737  043762          80$:  JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
5460  016242  004737  044046                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5461  016246  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5462  016250  000240                        NOP                     ;RETURN HERE IF ERROR
5463  016252  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5464  016254  000137  016550                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5465  016260  032737  100000  001340  90$:  BIT     #ATA,RMDS1              ;IS ATTENTION SET??
5466  016266  001012                        BNE     100$                    ;YES!!
```

```
5467  016270  012737  100000  001140          MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5468  016276  013737  001340  001142          MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5469  016304  042737  077777  001142          BIC     #^CATA,$BDDAT
5470  016312  104172                          ERROR   172                     ;REPORT ATA NOT SET
5471  016314                          100$:
5472  016314  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5473  016320  000000                          .WORD   0               ;MASK FOR RMER1
5474  016322  000000                          .WORD   0               ;MASK FOR RMER2
5475  016324  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5476  016326  000240                          NOP                     ;RETURN HERE IF ERROR
5477  016330  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5478  016332  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5479  016334  000137  016550                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5480  016340                          110$:
5481  016340  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5482  016346  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5483  016354  012737  000001  001376          MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
5484  016362  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5485  016366  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
5486  016370  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5487  016372  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5488  016374  000137  016550                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5489  016400                          120$:
5490  016400  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5491  016404  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5492  016406  000240                          NOP                     ;RETURN HERE IF ERROR
5493  016410  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5494  016412  000137  016550                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5495  016416                          130$:
5496  016416  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5497  016422  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
5498  016424  000240                          NOP                     ;RETURN HERE IF ERROR
5499  016426  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5500  016430  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5501  016432  000137  016550                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5502  016436  032737  100000  001340  140$:   BIT     #ATA,RMDSI              ;IS ATTENTION RESET??
5503  016444  001411                          BEQ     150$                    ;YES
5504  016446  013737  001340  001142          MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5505  016454  042737  077777  001142          BIC     #^CATA,$BDDAT
5506  016462  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
5507  016466  104246                          ERROR   246                     ;REPORT ATA NOT RESET
5508  016470                          150$:
5509  016470  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5510  016474  115760                          .WORD   NOTMSK          ;MASK FOR RMER1
5511  016476  000010                          .WORD   DPE             ;MASK FOR RMER2
5512  016500  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5513  016502  000240                          NOP                     ;RETURN HERE IF ERROR
5514  016504  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5515  016506  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5516  016510  000137  016550                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5517  016514                          160$:
5518  016514  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5519  016520  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
5520  016522  000240                          NOP                     ;RETURN HERE IF ERROR
5521  016524  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5522  016526  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
5523  016530  000137  016550                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5524  016534                            170$:
5525  016534  004737  045674                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5526  016540  000403                            BR      180$                    ;GO TO 180$ IF NO ERROR
5527  016542  000240                            NOP                     ;RETURN HERE IF ERROR
5528  016544  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5529  016546  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5530  016550                            180$:
5531
5532                                    ;;*****************************************************************
5533                                    ;*TEST 24       WRITE ATA TEST
5534                                    ;;*****************************************************************
5535  016550                            TST24:
5536  016550  000004                            SCOPE                   ;SCOPE CALL
5537  016552  000240                            NOP                     ;START OF TEST
5538  016554  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5539  016560  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5540  016564  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5541  016570  012737  000024  001226            MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5542
5543  016576  004737  043046                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5544  016602  054130                            .WORD   054130  ;TASK DESCRIPTOR
5545  016604  000404                            BR      70$                     ;GO TO 70$ IF NO ERROR
5546  016606  000240                            NOP                     ;RETURN HERE IF ERROR
5547  016610  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5548  016612  000137  017114                    JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5549  016616                            70$:
5550  016616  112737  000000  001533            MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5551  016624  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5552  016632  012737  000015  001376            MOV     #OFFSET!GO,RMCS1O       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5553  016640  004737  044316                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
5554  016644  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5555  016646  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
5556  016650  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5557  016652  000137  017114                    JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5558  016656  004737  043762            80$:    JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
5559  016662  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5560  016666  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5561  016670  000240                            NOP                     ;RETURN HERE IF ERROR
5562  016672  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5563  016674  000137  017114                    JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5564  016700  032737  100000  001340  90$:     BIT     #ATA,RMDSI              ;IS ATTENTION SET??
5565  016706  001012                            BNE     100$                    ;YES!!
5566  016710  013737  001340  001142            MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5567  016716  042737  077777  001142            BIC     #^CATA,$BDDAT
5568  016724  012737  100000  001140            MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5569  016732  104172                            ERROR   172                     ;REPORT ATA NOT SET
5570  016734  116102  000001            100$:   MOVB    1(R1),R2        ;R2 = ATTENTION BIT
5571  016740  042702  177400                    BIC     #^CATNMSK,R2    ;CLEAR SIGN EXTENSION
5572  016744  000240                            NOP
5573  016746  010237  001414                    MOV     R2,RMASO                ;PUT RMAS BIT IN OUTPUT BUF
5574  016752  112737  000016  001533            MOVB    #RMAS,PUTINX            ;WRITE REGISTER INDEX IN TABLE
5575  016760  112737  000200  001534            MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
5576  016766  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5577  016772  000404                            BR      110$    ;GO TO 110$ IF NO ERROR
5578  016774  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
5579  016776  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
5580  017000  000137  017114                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5581  017004                          110$:
5582  017004  004737  044046                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5583  017010  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
5584  017012  000240                          NOP                             ;RETURN HERE IF ERROR
5585  017014  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
5586  017016  000137  017114                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5587  017022                          120$:
5588  017022  004737  045042                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5589  017026  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
5590  017030  000240                          NOP                             ;RETURN HERE IF ERROR
5591  017032  104000                          ERROR#                          ;ERROR # DEFINED BY PRIERR SUBROUTINE
5592  017034  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5593  017036  000137  017114                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5594  017042  032737  100000  001340  130$:   BIT     #ATA,RMDSI              ;IS ATTENTION RESET??
5595  017050  001411                          BEQ     140$                    ;YES!!
5596  017052  013737  001340  001142          MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5597  017060  042737  077777  001142          BIC     #^CATA,$BDDAT
5598  017066  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
5599  017072  104247                          ERROR   247                     ;REPORT ATA NOT RESET
5600  017074                          140$:
5601  017074  004737  061416                  JSR     PC,CMPERRSTS            ;CHECK ANY ERRORS NOT MASKED
5602  017100  000000                          .WORD   0                       ;MASK FOR RMER1
5603  017102  000000                          .WORD   0                       ;MASK FOR RMER2
5604  017104  000403                          BR      150$                    ;GO TO 150$ IF NO ERROR
5605  017106  000240                          NOP                             ;RETURN HERE IF ERROR
5606  017110  104000                          ERROR                           ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5607  017112  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5608  017114                          150$:
5609
5610                                  ;;**************************************************************
5611                                  ;*TEST 25        ERROR/ATA TEST
5612                                  ;;**************************************************************
5613  017114                          TST25:
5614  017114  000004                          SCOPE                           ;SCOPE CALL
5615  017116  000240                          NOP                             ;START OF TEST
5616  017120  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
5617  017124  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5618  017130  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5619  017134  012737  000025  001226          MOV     #25,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5620
5621  017142  004737  043046                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5622  017146  054130                          .WORD   054130  ;TASK DESCRIPTOR
5623  017150  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5624  017152  000240                          NOP                             ;RETURN HERE IF ERROR
5625  017154  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5626  017156  000137  017506                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
5627  017162                          70$:
5628  017162  112737  000000  001533          MOVB    #RMCS1,PUTINX           ;SETUP PUT INDEX TABLE
5629  017170  112737  000200  001534          MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
5630  017176  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5631  017204  004737  044316                  JSR     PC,PUT                  ;CALL PUT SUBROUTINE
5632  017210  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5633  017212  000240                          NOP                             ;RETURN HERE TO REPORT AN ERROR
5634  017214  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
```

```
5635  017216  000137  017506                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5636  017222  004737  043762          80$:      JSR     PC,GETSTS               ;SETUP FOR STATUS
5637  017226  004737  044046                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5638  017232  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5639  017234  000240                            NOP                             ;RETURN HERE IF ERROR
5640  017236  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5641  017240  000137  017506                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5642  017244  032737  100000  001340  90$:      BIT     #ATA,RMDSI              ;IS ATA SET??
5643  017252  001012                            BNE     100$                    ;YES!!
5644  017254  013737  001340  001142            MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5645  017262  042737  077777  001142            BIC     #^CATA,$BDDAT
5646  017270  012737  100000  001140            MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5647  017276  104172                            ERROR   172                     ;REPORT ATA NOT SET
5648  017300                          100$:
5649  017300  112737  000014  001533            MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
5650  017306  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5651  017314  012737  040000  001412            MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
5652  017322  004737  044316                    JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
5653  017326  000404                            BR      110$    ;GO TO 110$ IF NO ERROR
5654  017330  000240                            NOP                     ;RETURN HERE TO REPORT ERROR
5655  017332  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5656  017334  000137  017506                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5657  017340                          110$:
5658  017340  112737  000000  001533            MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5659  017346  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5660  017354  012737  000001  001376            MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
5661  017362  004737  044316                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
5662  017366  000404                            BR      120$    ;GO TO 120$ IF NO ERROR
5663  017370  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
5664  017372  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5665  017374  000137  017506                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5666  017400                          120$:
5667  017400  004737  044046                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5668  017404  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
5669  017406  000240                            NOP                             ;RETURN HERE IF ERROR
5670  017410  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5671  017412  000137  017506                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5672  017416  032737  100000  001340  130$:     BIT     #ATA,RMDSI              ;IS ATA STILL SET??
5673  017424  001012                            BNE     140$                    ;YES!!
5674  017426  012737  100000  001140            MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5675  017434  013737  001340  001142            MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5676  017442  042737  077777  001142            BIC     #^CATA,$BDDAT
5677  017450  104250                            ERROR   250                     ;ATA SHOULD NOT RESET
5678  017452                          140$:
5679  017452  004737  060712                    JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5680  017456  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
5681  017460  000240                            NOP                     ;RETURN HERE IF ERROR
5682  017462  104000                            ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5683  017464  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5684  017466  000137  017506                    JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
5685  017472                          150$:
5686  017472  004737  045674                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5687  017476  000403                            BR      160$                    ;GO TO 160$ IF NO ERROR
5688  017500  000240                            NOP                     ;RETURN HERE IF ERROR
5689  017502  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5690  017504  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
5691  017506                                160$:
5692
5693                                        ;;*****************************************************************
5694                                        ;*TEST 26       PROGRAM INTERRUPT TEST
5695                                        ;;*****************************************************************
5696  017506                                TST26:
5697  017506  000004                              SCOPE                   ;SCOPE CALL
5698  017510  000240                              NOP                     ;START OF TEST
5699  017512  012706  001100                      MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5700  017516  013700  001276                      MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5701  017522  013701  001446                      MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5702  017526  012737  000026  001226              MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5703
5704  017534  004737  043046                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5705  017540  054130                              .WORD   054130  ;TASK DESCRIPTOR
5706  017542  000404                              BR      70$                     ;GO TO 70$ IF NO ERROR
5707  017544  000240                              NOP                     ;RETURN HERE IF ERROR
5708  017546  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5709  017550  000137  020154                      JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5710
5711  017554  113702  001272          70$:        MOVB    $VECT1,R2       ;R2 = RM03 INTERRUPT ADDRESS
5712  017560  042702  177400                      BIC     #^C<377>,R2     ;CLEAR SIGN EXTENSION
5713  017564  011204                              MOV     (R2),R4                 ;SAVE HANDLER ADDRESS
5714  017566  016205  000002                      MOV     2(R2),R5                ;SAVE HANDLER PRIORITY
5715  017572  012712  020026                      MOV     #150$,(R2)              ;WRITE HANDLER ADDRESS AND
5716  017576  012762  000300  000002              MOV     #PR6,2(R2)              ;PRIORITY FOR THIS TEST
5717  017604  113703  001273                      MOVB    $VECT1+1,R3     ;R3 = RM03 INTERRUPT LEVEL
5718  017610  042703  177400                      BIC     #^C<377>,R3     ;CLEAR SIGN EXTENSION
5719  017614  000303                              SWAB    R3
5720  017616  005303                              DEC     R3                      ;DECREMENT INTERRUPT LEVEL
5721  017620  100001                              BPL     80$
5722  017622  005003                              CLR     R3
5723  017624  042703  177437          80$:        BIC     #^CPR7,R3
5724  017630  000240                              NOP
5725  017632  012746  000300                      MOV     #PR6,-(SP)      ;;PUSH #PR6 ON STACK
5726  017636  012746  017764                      MOV     #120$,-(SP)     ;;PUSH #120$ ON STACK
5727  017642  010346                              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5728  017644  012746  017756                      MOV     #110$,-(SP)     ;;PUSH #110$ ON STACK
5729  017650  012737  177777  001414              MOV     #-1,RMAS0               ;WRITE ONES IN RMAS 10
5730  017656  112737  000016  001533              MOVB    #RMAS,PUTINX            ;CLEAR ALL ATTENTIONS
5731  017664  112737  000200  001533              MOVB    #200,PUTINX
5732  017672  004737  044316                      JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5733  017676  000402                              BR      90$     ;GO TO 90$ IF NO ERROR
5734  017700  000240                              NOP                     ;RETURN HERE IF ERROR
5735  017702  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
5736  017704                                90$:
5737  017704  112737  000000  001533              MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5738  017712  112737  000200  001534              MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5739  017720  012737  000115  001376              MOV     #OFFSET!GO!IE,RMCS10    ;RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
5740  017726  004737  044316                      JSR     PC,PUT          ;CALL PUT SUBROUTINE
5741  017732  000402                              BR      100$    ;GO TO 100$ IF NO ERROR
5742  017734  000240                              NOP                     ;RETURN HERE TO REPORT AN ERROR
5743  017736  104000                              ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5744  017740  004737  044656          100$:       JSR     PC,TIMOUT               ;WAIT FOR COMPLETION
5745  017744  004737  043762                      JSR     PC,GETSTS               ;SETUP FOR STATUS
5746  017750  012703  000205                      MOV     #205,R3                 ;R3=GROSS TIMEOUT
```

```
5747  017754  000002                          RTI                             ;DROP CP PRIORITY
5748  017756  005303          110$:           DEC     R3                      ;TIMEOUT THE INTERRUPT
5749  017760  100376                          BPL     110$
5750                          ;TIMEOUT BEFORE INTERRUPT
5751  017762  000002                          RTI                             ;RAISE CP PRIORITY
5752  017764                  120$:
5753  017764  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5754  017770  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5755  017772  000240                          NOP                     ;RETURN HERE IF ERROR
5756  017774  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5757  017776  000137  020154                  JMP     210$    ;GO TO 210$ IF ERROR WAS FOUND
5758  020002                  130$:
5759  020002  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5760  020006  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
5761  020010  000240                          NOP                     ;RETURN HERE IF ERROR
5762  020012  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5763  020014  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5764  020016  000137  020154                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5765  020022  104251          140$:           ERROR   251                     ;REPORT NO INTERRUPT
5766  020024  000423                          BR      180$
5767
5768  020026  022626          150$:           CMP     (SP)+,(SP)+             ;ADJUST STACK
5769  020030  012716  020036                  MOV     #160$,(SP)              ;CHANGE RETURN ADDRESS
5770  020034  000002                          RTI                             ;RAISE CP PRIORITY
5771  020036                  160$:
5772  020036  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5773  020042  000404                          BR      170$    ;GO TO 170$ IF NO ERROR
5774  020044  000240                          NOP                     ;RETURN HERE IF ERROR
5775  020046  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5776  020050  000137  020154                  JMP     210$    ;GO TO 210$ IF ERROR WAS FOUND
5777  020054                  170$:
5778  020054  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5779  020060  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5780  020062  000240                          NOP                     ;RETURN HERE IF ERROR
5781  020064  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5782  020066  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5783  020070  000137  020154                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5784  020074                  180$:
5785  020074  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5786  020100  115760                          .WORD   ND!MSK          ;MASK FOR RMER1
5787  020102  000010                          .WORD   DPE             ;MASK FOR RMER2
5788  020104  000405                          BR      190$                    ;GO TO 190$ IF NO ERROR
5789  020106  000240                          NOP                     ;RETURN HERE IF ERROR
5790  020110  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5791  020112  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5792  020114  000137  020154                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5793  020120                  190$:
5794  020120  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5795  020124  000405                          BR      200$                    ;GO TO 200$ IF NO ERROR
5796  020126  000240                          NOP                     ;RETURN HERE IF ERROR
5797  020130  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5798  020132  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5799  020134  000137  020154                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5800  020140                  200$:
5801  020140  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5802  020144  000403                          BR      210$                    ;GO TO 210$ IF NO ERROR
```

```
5803  020146  000240                        NOP                     ;RETURN HERE IF ERROR
5804  020150  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5805  020152  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5806  020154  010412                210$:   MOV     R4,(R2)
5807  020156  010562  000002              MOV     R5,2(R2)
5808                                  ;;**********************************************************************
5809                                  ;*TEST 27       INHIBIT INTERRUPT TEST
5810                                  ;;**********************************************************************
5811  020162                          TST27:
5812  020162  000004                        SCOPE                   ;SCOPE CALL
5813  020164  000240                        NOP                     ;START OF TEST
5814  020166  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5815  020172  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5816  020176  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5817  020202  012737  000027  001226        MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5818
5819  020210  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5820  020214  054130                        .WORD   054130  ;TASK DESCRIPTOR
5821  020216  000404                        BR      70$                     ;GO TO 70$ IF NO ERROR
5822  020220  000240                        NOP                     ;RETURN HERE IF ERROR
5823  020222  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5824  020224  000137  020514                JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5825  020230  113702  001272          70$:  MOVB    $VECT1,R2       ;R2 = RM03 INTERRUPT ADDRESS
5826  020234  042702  177400                BIC     #^C<377>,R2     ;CLEAR SIGN EXTENSION
5827  020240  011204                        MOV     (R2),R4                 ;SAVE HANDLER ADDRESS
5828  020242  016205  000002                MOV     2(R2),R5                ;SAVE HANDLER PRIORITY
5829  020246  012712  020460                MOV     #130$,(R2)              ;WRITE HANDLER ADDRESS AND
5830  020252  012762  000300  000002        MOV     #PR6,2(R2)              ;PRIORITY FOR THIS TEST
5831  020260  113703  001273                MOVB    $VECT1+1,R3     ;R3 = RM03 INTERRUPT LEVEL
5832  020264  042703  177400                BIC     #^C<377>,R3     ;CLEAR SIGN EXTENSION
5833  020270  012746  000300                MOV     #PR6,-(SP)      ;;PUSH #PR6 ON STACK
5834  020274  012746  020506                MOV     #180$,-(SP)     ;;PUSH #180$ ON STACK
5835  020300  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5836  020302  005316                        DEC     (SP)
5837  020304  100001                        BPL     80$
5838  020306  005016                        CLR     (SP)
5839  020310  042716  177437          80$:  BIC     #^CPR7,(SP)
5840  020314  012746  020446                MOV     #120$,-(SP)     ;;PUSH #120$ ON STACK
5841  020320  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5842  020322  012746  020400                MOV     #100$,-(SP)     ;;PUSH #100$ ON STACK
5843  020326  004737  043762                JSR     PC,GETSTS               ;SETUP FOR STATUS
5844  020332  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5845  020340  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5846  020346  012737  000115  001376        MOV     #OFFSET!GO!IE,RMCS10    ;RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
5847  020354  004737  044316                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5848  020360  000402                        BR      90$     ;GO TO 90$ IF NO ERROR
5849  020362  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
5850  020364  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5851  020366  004737  044656          90$:  JSR     PC,TIMOUT               ;WAIT FOR GO=0
5852  020372  012703  000205                MOV     #205,R3                 ;R3=GROSS TIMER
5853  020376  000002                        RTI                             ;MAKE PRIORITY=RM01
5854  020400  005303                100$:   DEC     R3                      ;DELAY
5855  020402  100376                        BPL     100$
5856  020404  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5857  020412  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5858  020420  012737  000000  001376        MOV     #NOP,RMCS10     ;RMCS1 OUTPUT BUFFER = NOP
```

```
5859  020426  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5860  020432  000402                          BR      110$    ;GO TO 110$ IF NO ERROR
5861  020434  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5862  020436  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5863  020440  012703  000205          110$:   MOV     #205,R3
5864  020444  000002                          RTI
5865  020446  012712  020462          120$:   MOV     #140$,(R2)              ;MAKE PRIORITY < RM01
5866  020452  005303                  125$:   DEC     R3
5867  020454  100376                          BPL     125$
5868  020456  000002                          RTI                             ;NO ERROR
5869  020460  022626                  130$:   CMP     (SP)+,(SP)+             ;ADJUST STACK
5870  020462  022626                  140$:   CMP     (SP)+,(SP)+             ;ADJUST STACK
5871  020464  012716  020472                  MOV     #160$,(SP)
5872  020470  000002                          RTI
5873  020472                          160$:
5874  020472  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5875  020476  000402                          BR      170$    ;GO TO 170$ IF NO ERROR
5876  020500  000240                          NOP                     ;RETURN HERE IF ERROR
5877  020502  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5878  020504  104252                  170$:   ERROR   252                     ;SHOULD NOT HAVE INTERRUPT
5879  020506  010412                  180$:   MOV     R4,(R2)
5880  020510  010562  000002                  MOV     R5,2(R2)
5881  020514                          210$:
5882                                  ;;*****************************************************************
5883                                  ;*TEST 30       RETURN TO CENTERLINE TEST
5884                                  ;;*****************************************************************
5885  020514                          TST30:
5886  020514  000004                          SCOPE                   ;SCOPE CALL
5887  020516  000240                          NOP                     ;START OF TEST
5888  020520  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5889  020524  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5890  020530  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5891  020534  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5892  020542  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5893  020546  054130                          .WORD   054130  ;TASK DESCRIPTOR
5894  020550  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5895  020552  000240                          NOP                     ;RETURN HERE IF ERROR
5896  020554  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5897  020556  000137  021036                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5898  020562                          70$:
5899  020562  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5900  020570  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5901  020576  012737  000017  001376          MOV     #RTC!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = RTC!GO
5902  020604  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5903  020610  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5904  020612  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5905  020614  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5906  020616  000137  021036                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5907  020622  004737  043762          80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5908  020626  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
5909  020632  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5910  020636  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5911  020640  000240                          NOP                     ;RETURN HERE IF ERROR
5912  020642  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5913  020644  000137  021036                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5914  020650                          90$:
```

```
5915  020650  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5916  020654  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
5917  020656  000240                  NOP                     ;RETURN HERE IF ERROR
5918  020660  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5919  020662  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5920  020664  000137  021036          JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5921  020670  032737  000001  001340  100$:   BIT     #OM,RMDSI       ;OFFSET MODE OFF??
5922  020676  001411                  BEQ     110$            ;YES!!
5923  020700  005037  001140          CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
5924  020704  013737  001340  001142  MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
5925  020712  042737  177776  001142  BIC     #^COM,$BDDAT
5926  020720  104157                  ERROR   157             ;OFFSET MODE NOT RESET
5927  020722  032737  100000  001340  110$:   BIT     #ATA,RMDSI      ;WAS ATTENTION SET ??
5928  020730  001012                  BNE     120$            ;YES!!
5929  020732  013737  001340  001142  MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
5930  020740  042737  077777  001142  BIC     #^CATA,$BDDAT
5931  020746  012737  100000  001140  MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
5932  020754  104171                  ERROR   171             ;"ATA" NOT SET DURING RTC
5933  020756                  120$:
5934  020756  004737  061416          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5935  020762  115760                  .WORD   NOTMSK          ;MASK FOR RMER1
5936  020764  000010                  .WORD   DPE             ;MASK FOR RMER2
5937  020766  000405                  BR      130$                    ;GO TO 130$ IF NO ERROR
5938  020770  000240                  NOP                     ;RETURN HERE IF ERROR
5939  020772  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5940  020774  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5941  020776  000137  021036          JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5942  021002                  130$:
5943  021002  004737  060712          JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5944  021006  000405                  BR      140$                    ;GO TO 140$ IF NO ERROR
5945  021010  000240                  NOP                     ;RETURN HERE IF ERROR
5946  021012  104000                  ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5947  021014  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5948  021016  000137  021036          JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5949  021022                  140$:
5950  021022  004737  045674          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5951  021026  000403                  BR      150$                    ;GO TO 150$ IF NO ERROR
5952  021030  000240                  NOP                     ;RETURN HERE IF ERROR
5953  021032  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5954  021034  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5955  021036                  150$:
5956
5957                          ;;*****************************************************************
5958                          ;*TEST 31        READ IN PRESET TEST
5959
5960                          ;;*****************************************************************
5961  021036                  TST31:
5962  021036  000004                  SCOPE                   ;SCOPE CALL
5963  021040  000240                  NOP                     ;START OF TEST
5964  021042  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5965  021046  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5966  021052  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5967  021056  012737  000031  001226  MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5968
5969  021064  004737  043046          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5970  021070  054130                  .WORD   054130  ;TASK DESCRIPTOR
```

```
5971  021072  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5972  021074  000240                          NOP                     ;RETURN HERE IF ERROR
5973  021076  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5974  021100  000137  021420                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
5975  021104                          70$:
5976  021104  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
5977  021110  112722  000006                  MOVB    #RMDA,(R2)+
5978  021114  112722  000034                  MOVB    #RMDC,(R2)+
5979  021120  112722  000032                  MOVB    #RMOF,(R2)+
5980  021124  112722  000000                  MOVB    #RMCS1,(R2)+
5981  021130  112722  000200                  MOVB    #200,(R2)+
5982  021134  012737  177777  001404          MOV     #-1,RMDA0
5983  021142  012737  177777  001432          MOV     #-1,RMDC0
5984  021150  012737  016200  001430          MOV     #FMT16!ECI!HCI!OFD,RMOF0
5985  021156  012737  000021  001376          MOV     #RIP!GO,RMCS10
5986  021164  004737  044316                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5987  021170  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5988  021172  000240                          NOP                     ;RETURN HERE IF ERROR
5989  021174  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5990  021176  000137  021420                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5991  021202  004737  043762          80$:    JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
5992  021206  004737  044656                  JSR     PC,TIMOUT               ;WAIT FOR RIP TO COMPLETE
5993  021212  004737  044046                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5994  021216  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5995  021220  000240                          NOP                     ;RETURN HERE IF ERROR
5996  021222  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5997  021224  000137  021420                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5998  021230                          90$:
5999  021230  004737  045042                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6000  021234  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6001  021236  000240                          NOP                     ;RETURN HERE IF ERROR
6002  021240  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6003  021242  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6004  021244  000137  021420                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
6005  021250  013737  001360  001142  100$:   MOV     RMOFI,$BDDAT    ;IS RMOF RESET??
6006  021256  042737  161577  001142          BIC     #^C<FMT16!ECI!HCI!OFD>,$BDDAT
6007  021264  001403                          BEQ     110$                    ;YES!!
6008  021266  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
6009  021272  104160                          ERROR   160                     ;RMOF NOT CLEARED BY RIP
6010  021274  005737  001334          110$:   TST     RMDAI                   ;IS RMDA RESET??
6011  021300  001406                          BEQ     120$                    ;YES!!
6012  021302  013737  001334  001142          MOV     RMDAI,$BDDAT    ;BAD DATA FOR TYPEOUT
6013  021310  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
6014  021314  104161                          ERROR   161                     ;RMDA NOT CLEARED BY RIP
6015  021316  005737  001362          120$:   TST     RMDCI                   ;IS RMDC RESET??
6016  021322  001406                          BEQ     130$                    ;YES!!
6017  021324  013737  001362  001142          MOV     RMDCI,$BDDAT    ;BAD DATA FOR TYPEOUT
6018  021332  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
6019  021336  104162                          ERROR   162                     ;RMDC NOT CLEARED BY RIP
6020  021340                          130$:
6021  021340  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
6022  021344  115760                          .WORD   ND!MSK                  ;MASK FOR RMER1
6023  021346  000010                          .WORD   DPE                     ;MASK FOR RMER2
6024  021350  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
6025  021352  000240                          NOP                     ;RETURN HERE IF ERROR
6026  021354  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
```

```
6027  021356  004736                          T31     READ IN PRESET TEST
6028  021360  000137  021420                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6029  021364                                            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
6030  021364  004737  060712                    140$:
6031  021370  000405                                    JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
6032  021372  000240                                    BR      150$                    ;GO TO 150$ IF NO ERROR
6033  021374  104000                                    NOP                     ;RETURN HERE IF ERROR
6034  021376  004736                                    ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
6035  021400  000137  021420                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6036  021404                                            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
6037  021404  004737  045674                    150$:
6038  021410  000403                                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6039  021412  000240                                    BR      160$                    ;GO TO 160$ IF NO ERROR
6040  021414  104000                                    NOP                     ;RETURN HERE IF ERROR
6041  021416  004736                                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6042  021420                                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6043                                            160$:
6044                                            ;;*****************************************************************
6045                                            ;*TEST 32        RMDC CLEAR OFFSET TEST
6046                                            ;;*****************************************************************
6047  021420                                    TST32:
6048  021420  000004                                    SCOPE                   ;SCOPE CALL
6049  021422  000240                                    NOP                     ;START OF TEST
6050  021424  012706  001100                            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6051  021430  013700  001276                            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6052  021434  013701  001446                            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6053  021440  012737  000032  001226                    MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6054
6055  021446  004737  043046                            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6056  021452  054130                                    .WORD   054130  ;TASK DESCRIPTOR
6057  021454  000404                                    BR      70$                     ;GO TO 70$ IF NO ERROR
6058  021456  000240                                    NOP                     ;RETURN HERE IF ERROR
6059  021460  104000                                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6060  021462  000137  021736                            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
6061  021466                                    70$:
6062  021466  112737  000000  001533                    MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6063  021474  112737  000200  001534                    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6064  021502  012737  000015  001376                    MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
6065  021510  004737  044316                            JSR     PC,PUT          ;CALL PUT SUBROUTINE
6066  021514  000404                                    BR      80$     ;GO TO 80$ IF NO ERROR
6067  021516  000240                                    NOP                     ;RETURN HERE TO REPORT AN ERROR
6068  021520  104000                                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6069  021522  000137  021736                            JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
6070  021526  004737  043762                    80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS
6071  021532  004737  044046                            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6072  021536  000404                                    BR      90$     ;GO TO 90$ IF NO ERROR
6073  021540  000240                                    NOP                     ;RETURN HERE IF ERROR
6074  021542  104000                                    ERROR                   ;ERROR DEFINED BY GET SUB
6075  021544  000137  021736                            JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
6076  021550                                    90$:
6077  021550  004737  045042                            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6078  021554  000405                                    BR      100$                    ;GO TO 100$ IF NO ERROR
6079  021556  000240                                    NOP                     ;RETURN HERE IF ERROR
6080  021560  104000                                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6081  021562  004736                                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6082  021564  000137  021736                            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
```

```
6083  021570  032737  000001  001340  100$:   BIT     #OM,RMDSI       ;OFFSET ON??
6084  021576  001013                          BNE     110$            ;YES
6085  021600  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
6086  021606  042737  177776  001142          BIC     #^COM,$BDDAT
6087  021614  012737  000001  001140          MOV     #OM,$GDDAT      ;GOOD DATA FOR TYPEOUT
6088  021622  104156                          ERROR   156             ;OFFSET MODE NOT SET
6089  021624  000444                          BR      140$            ;SKIP REST OF TEST
6090  021626                          110$:
6091  021626  012737  000000  001432          MOV     #0,RMDCO        ;CYLINDER = 0
6092  021634  112737  000034  001533          MOVB    #RMDC,PUTINX            ;SETUP REGISTER INDEX TABLE
6093  021642  112737  000200  001534          MOVB    #200,PUTINX+1           ;TERMINATE TABLE
6094  021650  004737  044316                  JSR     PC,PUT                  ;WRITE RMDC USING PUT SUB
6095  021654  000404                          BR      120$            ;GO TO 120$ IF NO ERROR
6096  021656  000240                          NOP                     ;RETURN HERE IF ERROR
6097  021660  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT
6098  021662  000137  021736                  JMP     140$                    ;GO TO 140$ IF ERROR
6099  021666                          120$:
6100  021666  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6101  021672  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6102  021674  000240                          NOP                     ;RETURN HERE IF ERROR
6103  021676  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6104  021700  000137  021736                  JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
6105  021704  032737  000001  001340  130$:   BIT     #OM,RMDSI       ;DID OFFSET MODE RESET??
6106  021712  001411                          BEQ     140$
6107  021714  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
6108  021722  042737  177776  001142          BIC     #^COM,$BDDAT
6109  021730  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
6110  021734  104253                          ERROR   253             ;WRITING RMDC DIDNT CLEAR OM
6111  021736                          140$:
6112
6113                                  ;;*****************************************************************
6114                                  ;*TEST 33       ILLEGAL FUNCTION TEST
6115                                  ;;*****************************************************************
6116  021736                          TST33:
6117  021736  000004                          SCOPE                   ;SCOPE CALL
6118  021740  000240                          NOP                     ;START OF TEST
6119  021742  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6120  021746  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6121  021752  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6122  021756  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6123
6124  021764  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6125  021770  040000                          .WORD   040000  ;TASK DESCRIPTOR
6126  021772  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
6127  021774  000240                          NOP                     ;RETURN HERE IF ERROR
6128  021776  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6129  022000  000137  022522                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6130  022004  004737  043762          5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
6131  022010  012702  000000                  MOV     #NOP,R2
6132  022014                          10$:
6133  022014  004737  053210                  JSR     PC,CNTCLR
6134  022020  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
6135  022022  000240                          NOP                     ;RETURN HERE IF ERROR
6136  022024  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
6137  022026  000137  022522                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6138  022032                          20$:
```

```
                                        ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
                                        ;CYLINDER
022032  112737  000024  001533            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
022040  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
022046  012737  000001  001422            MOV     #DMD,RMMR10     ;RMMR1=DMD
022054  004737  044316                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
022060  000404                            BR      30$             ;GO TO 30$ IF NO ERROR
022062  000240                            NOP                     ;RETURN HERE IF ERROR
022064  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
022066  000137  022522                    JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
022072                                  30$:
022072  112737  000024  001533            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
022100  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
022106  012737  001401  001422            MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
022114  004737  044316                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
022120  000404                            BR      35$             ;GO TO 35$ IF NO ERROR
022122  000240                            NOP                     ;RETURN HERE IF ERROR
022124  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
022126  000137  022522                    JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
022132                                  35$:

                                        ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
022132  112737  000000  001533            MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
022140  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
022146  012737  000023  001376            MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
022154  004737  044316                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
022160  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
022162  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
022164  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
022166  000137  022522                    JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
022172                                  40$:
022172  012737  000001  00137             MOV     #GO,RMCS10
022200  050237  001376                    BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
022204  012737  105642  001402            MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
022212  012737  177777  001400            MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
022220  005037  001404                    CLR     RMDA0           ;CLEAR DISK ADDRESS
022224  005037  001432                    CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
022230  012737  010000  001430            MOV     #FMT16,RMOF0    ;16 BHT FORMAT
022236  012703  001533                    MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
022242  112723  000004                    MOVB    #RMBA,(R3)+
022246  112723  000002                    MOVB    #RMWC,(R3)+
022252  112723  000032                    MOVB    #RMOF,(R3)+
022256  112723  000006                    MOVB    #  MDA,(R3)+
022262  112723  000034                    MOVB    #RMDC,(R3)+
022266  112723  000000                    MOVB    #RMCS1,(R3)+
022272  112713  000200                    MOVB    #200,(R3)
022276  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
022302  000404                            BR      50$     ;GO TO 50$ IF NO ERROR
022304  000240                            NOP                     ;RETURN HERE IF ERROR
022306  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
022310  000137  022522                    JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
022314  004737  044656                  50$:      JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
022320  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
022324  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
022326  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
                                        ERROR                           ;ERROR DEFINED BY GET SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                60$:
                                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                                        BR      70$                             ;GO TO 70$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     110$                            ;GO TO 110$ IF ERROR WAS FOUND
                                70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
                                        BIC     #^CILF,$GDDAT
                                        MOV     #MER11,$BDDAT   ;BAD DATA FOR TYPEOUT
                                        BIC     #^CILF,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS ILF STATUS CORRECT?
                                        BEQ     80$             ;YES!!
                                        ERROR   254             ;REPORT INCORRECT ILF STATUS
                                        BR      110$
                                80$:    MOV     FNCDTB(R2),$GDDAT
                                        BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
                                        BEQ     90$             ;NO!!
                                        BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
                                90$:    BIC     #^CATA,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
                                        BIC     #^CATA,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
                                        BEQ     100$            ;YES!!
                                        ERROR   255             ;INCORRECT ATA STATUS
                                        BR      110$
                                100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
                                        CMP     #ILF76,R2       ;DONE??
                                        BLO     110$            ;YES !!
                                        JMP     10$             ;TEST NEXT FUNCTION
                                110$:
                                ;;*****************************************************************
                                ;*TEST 34        INVALID COMMAND TEST
                                ;;*****************************************************************
                                TST34:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #34,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                                        .WORD   040000  ;TASK DESCRIPTOR
                                        BR      5$                              ;GO TO 5$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     110$                            ;GO TO 110$ IF ERROR WAS FOUND
                                5$:     JSR     PC,GETSTS               ;SETUP FOR STATUS
                                        MOV     #NOP,R2
                                10$:
                                        JSR     PC,CNTCLR
                                        BR      20$     ;GO TO 20$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
```

```
                                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                20$:

                                ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
                                ;CYLINDER
                                        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
                                        MOV     #DMD,RMMR10     ;RMMR1=DMD
                                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
                                        BR      30$             ;GO TO 30$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                                        JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
                                30$:
                                        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
                                        MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
                                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
                                        BR      35$             ;GO TO 35$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                                        JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
                                35$:


                                ;VOLUME VALID IS LEFT RESET FOR THE TEST

                                40$:
                                        MOV     #GO,RMCS10
                                        BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
                                        MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
                                        MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
                                        CLR     RMDA0           ;CLEAR DISK ADDRESS
                                        CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
                                        MOV     #FMT16,RMOF0    ;16 BHT FORMAT
                                        MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
                                        MOVB    #RMBA,(R3)+
                                        MOVB    #RMWC,(R3)+
                                        MOVB    #RMOF,(R3)+
                                        MOVB    #RMDA,(R3)+
                                        MOVB    #RMDC,(R3)+
                                        MOVB    #RMCS1,(R3)+
                                        MOVB    #200,(R3)
                                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
                                        BR      50$     ;GO TO 50$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR DEFINED BY PUT SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
                                        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
                                        BR      60$     ;GO TO 60$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR DEFINED BY GET SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                60$:
```

```
6307  023062  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6308  023066  000405                  BR      70$                     ;GO TO 70$ IF NO ERROR
6309  023070  000240                  NOP                     ;RETURN HERE IF ERROR
6310  023072  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6311  023074  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6312  023076  000137  023246          JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6313  023102  016237  066770  001140 70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
6314  023110  042737  167777  001140        BIC     #^tCIVC,$GDDAT
6315  023116  013737  001370  001142        MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
6316  023124  042737  167777  001142        BIC     #^tCIVC,$BDDAT
6317  023132  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS IVC STATUS CORRECT?
6318  023140  001402                  BEQ     80$             ;YES!!
6319  023142  104216                  ERROR   216             ;REPORT INCORRECT IVC STATUS
6320  023144  000440                  BR      110$
6321  023146  016237  066770  001140 80$:    MOV     FNCDTB(R2),$GDDAT
6322  023154  032737  040000  001340        BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
6323  023162  001403                  BEQ     90$             ;NO!!
6324  023164  052737  100000  001140        BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
6325  023172  042737  077777  001140 90$:    BIC     #^tCATA,$GDDAT
6326  023200  013737  001340  001142        MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
6327  023206  042737  077777  001142        BIC     #^tCATA,$BDDAT
6328  023214  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6329  023222  001402                  BEQ     100$            ;YES!!
6330  023224  104255                  ERROR   255             ;INCORRECT ATA STATUS
6331  023226  000407                  BR      110$
6332  023230  062702  000002 100$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
6333  023234  022702  000076                  CMP     #ILF76,R2       ;DONE??
6334  023240  103402                  BLO     110$            ;YES !!
6335  023242  000137  022600                  JMP     10$             ;TEST NEXT FUNCTION
6336  023246                  110$:
6337                          ;;*****************************************************************
6338                          ;*TEST 35       INVALID ADDRESS ERROR TEST
6339                          ;;*****************************************************************
6340  023246                  T35:
6341  023246  000004                  SCOPE                   ;SCOPE CALL
6342  023250  000240                  NOP                     ;START OF TEST
6343  023252  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6344  023256  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6345  023262  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6346  023266  012737  000035  001226  MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6347
6348  023274  004737  043046          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6349  023300  040000                  .WORD   040000  ;TASK DESCRIPTOR
6350  023302  000404                  BR      5$                      ;GO TO 5$ IF NO ERROR
6351  023304  000240                  NOP                     ;RETURN HERE IF ERROR
6352  023306  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6353  023310  000137  024036          JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6354  023314  004737  043762  5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
6355  023320  012702  000000          MOV     #NOP,R2
6356  023324                  10$:
6357  023324  004737  053210          JSR     PC,CNTCLR
6358  023330  000404                  BR      20$     ;GO TO 20$ IF NO ERROR
6359  023332  000240                  NOP                     ;RETURN HERE IF ERROR
6360  023334  104000                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
6361  023336  000137  024036          JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6362  023342                  20$:
```

```
                                        ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
                                        ;CYLINDER
023342  112737  000024  001533            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
023350  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
023356  012737  000001  001422            MOV     #DMD,RMMR10     ;RMMR1=DMD
023364  004737  044316                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
023370  000404                            BR      30$             ;GO TO 30$ IF NO ERROR
023372  000240                            NOP                     ;RETURN HERE IF ERROR
023374  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
023376  000137  024036                    JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
023402                                  30$:
023402  112737  000024  001533            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
023410  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
023416  012737  001401  001422            MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
023424  004737  044316                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
023430  000404                            BR      35$             ;GO TO 35$ IF NO ERROR
023432  000240                            NOP                     ;RETURN HERE IF ERROR
023434  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
023436  000137  024036                    JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
023442                                  35$:

                                        ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
023442  112737  000000  001533            MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
023450  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
023456  012737  000023  001376            MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
023464  004737  044316                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
023470  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
023472  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
023474  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
023476  000137  024036                    JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
023502                                  40$:
023502  012737  000001  001376            MOV     #GO,RMCS10
023510  050237  001376                    BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
023514  012737  105642  001402            MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
023522  012737  177777  001400            MOV     #^C1+1,RMWC0    ;DUMMY WORD COUNT
023530  012737  177777  001404            MOV     #-1,RMDA0       ;USE INVALID DISK ADDRESS
023536  012737  177777  001432            MOV     #-1,RMDC0       ;USE INVALID CYLINDER ADDRESS
023544  012737  010000  001430            MOV     #FMT16,RMOF0    ;16 BIT FORMAT
023552  012703  001533                    MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
023556  112723  000004                    MOVB    #RMBA,(R3)+
023562  112723  000002                    MOVB    #RMWC,(R3)+
023566  112723  000032                    MOVB    #RMOF,(R3)+
023572  112723  000006                    MOVB    #RMDA,(R3)+
023576  112723  000034                    MOVB    #RMDC,(R3)+
023602  112723  000000                    MOVB    #RMCS1,(R3)+
023606  112713  000200                    MOVB    #200,(R3)
023612  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
023616  000404                            BR      50$     ;GO TO 50$ IF NO ERROR
023620  000240                            NOP                     ;RETURN HERE IF ERROR
023622  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
023624  000137  024036                    JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
023630  004737  044656                  50$:  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
023634  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
023640  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
023642  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
                                ERROR                           ;ERROR DEFINED BY GET SUB
                                JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                        60$:
                                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                                BR      70$                             ;GO TO 70$ IF NO ERROR
                                NOP                             ;RETURN HERE IF ERROR
                                ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                JMP     110$                            ;GO TO 110$ IF ERROR WAS FOUND
                        70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
                                BIC     #^CIAE,$GDDAT
                                MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
                                BIC     #^CIAE,$BDDAT
                                CMP     $GDDAT,$BDDAT   ;IS IAE STATUS CORRECT?
                                BEQ     80$             ;YES!!
                                ERROR   217             ;REPORT INCORRECT IAE STATUS
                                BR      110$
                        80$:    MOV     FNCDTB(R2),$GDDAT
                                BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
                                BEQ     90$             ;NO!!
                                BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
                        90$:    BIC     #^CATA,$GDDAT
                                MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
                                BIC     #^CATA,$BDDAT
                                CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
                                BEQ     100$            ;YES!!
                                ERROR   255             ;INCORRECT ATA STATUS
                                BR      110$
                        100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
                                CMP     #ILF76,R2       ;DONE??
                                BLO     110$            ;YES !!
                                JMP     10$             ;TEST NEXT FUNCTION
                        110$:
                        ;;*****************************************************************
                        ;*TEST 36       WRITE LOCK ERROR TEST
                        ;;*****************************************************************
                        TST36:
                                SCOPE                           ;SCOPE CALL
                                NOP                             ;START OF TEST
                                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                MOV     #36,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

                                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                                .WORD   040000  ;TASK DESCRIPTOR
                                BR      5$                              ;GO TO 5$ IF NO ERROR
                                NOP                             ;RETURN HERE IF ERROR
                                ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                JMP     110$                            ;GO TO 110$ IF ERROR WAS FOUND
                        5$:     JSR     PC,GETSTS               ;SETUP FOR STATUS
                                MOV     #NOP,R2
                        10$:
                                JSR     PC,CNTCLR
                                BR      20$     ;GO TO 20$ IF NO ERROR
                                NOP                             ;RETURN HERE IF ERROR
```

```
                                T36     WRITE LOCK ERROR TEST

                                        ERROR                   ;ERROR NUMBER DEFINED BY SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                20$:

                                ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
                                ;CYLINDER, AND DIAGNOSTIC WRITE LOCK
  6481  024132  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
  6482  024140  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
  6483  024146  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
  6484  024154  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
  6485  024160  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
  6486  024162  000240                          NOP                     ;RETURN HERE IF ERROR
  6487  024164  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
  6488  024166  000137  024622                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
  6489  024172                        30$:
  6490  024172  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
  6491  024200  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
  6492  024206  012737  001411  001422          MOV     #DMD!MUR!MOC!MWP,RMMR10 ;RMMR1=DMD!MUR!MOC!MWP
  6493  024214  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
  6494  024220  000404                          BR      35$             ;GO TO 35$ IF NO ERROR
  6495  024222  000240                          NOP                     ;RETURN HERE IF ERROR
  6496  024224  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
  6497  024226  000137  024622                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
  6498  024232                        35$:
  6499
  6500                                  ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
  6501  024232  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
  6502  024240  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
  6503  024246  012737  000023  001376          MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
  6504  024254  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
  6505  024260  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
  6506  024262  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
  6507  024264  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
  6508  024266  000137  024622                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
  6509  024272                        40$:
  6510  024272  012737  000001  001376          MOV     #GO,RMCS10
  6511  024300  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
  6512  024304  012737  105642  001402          MOV     #BUFONE,RMBA0   ,DUMMY BUS ADDRESS
  6513  024312  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
  6514  024320  005037  001404                  CLR     RMDA0           ;CLEAR DISK ADDRESS
  6515  024324  005037  001432                  CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
  6516  024330  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BHT FORMAT
  6517  024336  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
  6518  024342  112723  000004                  MOVB    #RMBA,(R3)+
  6519  024346  112723  000002                  MOVB    #RMWC,(R3)+
  6520  024352  112723  000032                  MOVB    #RMOF,(R3)+
  6521  024356  112723  000006                  MOVB    #RMDA,(R3)+
  6522  024362  112723  000034                  MOVB    #RMDC,(R3)+
  6523  024366  112723  000000                  MOVB    #RMCS1,(R3)+
  6524  024372  112713  000200                  MOVB    #200,(R3)
  6525  024376  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
  6526  024402  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
  6527  024404  000240                          NOP                     ;RETURN HERE IF ERROR
  6528  024406  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
  6529  024410  000137  024622                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
  6530  024414  004737  044656          50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
```

```
6531  024420  004737  044046          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6532  024424  000404                  BR      60$     ;GO TO 60$ IF NO ERROR
6533  024426  000240                  NOP                     ;RETURN HERE IF ERROR
6534  024430  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
6535  024432  000137  024622          JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6536  024436                  60$:
6537  024436  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6538  024442  000405                  BR      70$                     ;GO TO 70$ IF NO ERROR
6539  024444  000240                  NOP                     ;RETURN HERE IF ERROR
6540  024446  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6541  024450  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6542  024452  000137  024622          JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6543  024456  016237  066770  001140  70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
6544  024464  042737  173777  001140          BIC     #^CWLE,$GDDAT
6545  024472  013737  001342  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
6546  024500  042737  173777  001142          BIC     #^CWLE,$BDDAT
6547  024506  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS WLE STATUS CORRECT?
6548  024514  001402                  BEQ     80$             ;YES!!
6549  024516  104220                  ERROR   220             ;REPORT INCORRECT WLE STATUS
6550  024520  000440                  BR      110$
6551  024522  016237  066770  001140  80$:    MOV     FNCDTB(R2),$GDDAT
6552  024530  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
6553  024536  001403                  BEQ     90$             ;NO!!
6554  024540  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
6555  024546  042737  077777  001140  90$:    BIC     #^CATA,$GDDAT
6556  024554  013737  001340  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
6557  024562  042737  077777  001142          BIC     #^CATA,$BDDAT
6558  024570  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6559  024576  001402                  BEQ     100$            ;YES!!
6560  024600  104255                  ERROR   255             ;INCORRECT ATA STATUS
6561  024602  000407                  BR      110$
6562  024604  062702  000002  100$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
6563  024610  022702  000076          CMP     #ILF76,R2       ;DONE??
6564  024614  103402                  BLO     110$            ;YES !!
6565  024616  000137  024114          JMP     10$             ;TEST NEXT FUNCTION
6566  024622                  110$:
6567                          ;;*****************************************************************
6568                          ;*TEST 37       OPI TEST
6569                          ;;*****************************************************************
6570  024622                  TST37:
6571  024622  000004                  SCOPE                   ;SCOPE CALL
6572  024624  000240                  NOP                     ;START OF TEST
6573  024626  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6574  024632  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6575  024636  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6576  024642  012737  000037  001226  MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6577
6578  024650  004737  043046          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6579  024654  040000                  .WORD   040000  ;TASK DESCRIPTOR
6580  024656  000404                  BR      5$                      ;GO TO 5$ IF NO ERROR
6581  024660  000240                  NOP                     ;RETURN HERE IF ERROR
6582  024662  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6583  024664  000137  025446          JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6584  024670  004737  043762  5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
6585  024674  012702  000000          MOV     #NOP,R2
6586  024700                  10$:
```

```
6587  024700  004737  053210                        JSR     PC,CNTCLR
6588  024704  000404                                BR      20$     ;GO TO 20$ IF NO ERROR
6589  024706  000240                                NOP                     ;RETURN HERE IF ERROR
6590  024710  104000                                ERROR                   ;ERROR NUMBER DEFINED BY SUB
6591  024712  000137  025446                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6592  024716                                20$:
6593
6594                                        ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
6595                                        ;CYLINDER
6596  024716  112737  000024  001533                MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6597  024724  112737  000200  001534                MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6598  024732  012737  000001  001422                MOV     #DMD,RMMR10     ;RMMR1=DMD
6599  024740  004737  044316                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6600  024744  000404                                BR      30$             ;GO TO 30$ IF NO ERROR
6601  024746  000240                                NOP                     ;RETURN HERE IF ERROR
6602  024750  104000                                ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6603  024752  000137  025446                        JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6604  024756                                30$:
6605  024756  112737  000024  001533                MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6606  024764  112737  00020..  001534               MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6607  024772  012737  001401  001422                MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6608  025000  004737  044316                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6609  025004  000404                                BR      35$             ;GO TO 35$ IF NO ERROR
6610  025006  000240                                NOP                     ;RETURN HERE IF ERROR
6611  025010  104000                                ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6612  025012  000137  025446                        JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6613  025016                                35$:
6614
6615                                        ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
6616  025016  112737  000000  001533                MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6617  025024  112737  000200  001534                MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6618  025032  012737  000023  001376                MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6619  025040  004737  044316                        JSR     PC,PUT          ;CALL PUT SUBROUTINE
6620  025044  000404                                BR      36$     ;GO TO 36$ IF NO ERROR
6621  025046  000240                                NOP                     ;RETURN HERE TO REPORT AN ERROR
6622  025050  104000                                ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6623  025052  000137  025446                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6624  025056                                36$:
6625  025056  112737  000024  001533                MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6626  025064  112737  000200  001534                MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6627  025072  012737  000401  001422                MOV     #DMD!MOC,RMMR10 ;RMMR1=DMD!MOC
6628  025100  004737  044316                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6629  025104  000404                                BR      40$             ;GO TO 40$ IF NO ERROR
6630  025106  000240                                NOP                     ;RETURN HERE IF ERROR
6631  025110  104000                                ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6632  025112  000137  025446                        JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6633  025116                                40$:
6634  025116  012737  000001  001376                MOV     #GO,RMCS10
6635  025124  050237  001376                        BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6636  025130  012737  105642  001402                MOV     #BUFOME,RMBA0   ;DUMMY BUS ADDRESS
6637  025136  012737  177777  001400                MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6638  025144  005037  001404                        CLR     RMDA0           ;CLEAR DISK ADDRESS
6639  025150  005037  001432                        CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
6640  025154  012737  010000  001430                MOV     #FMT16,RMOF0    ;16 BHT FORMAT
6641  025162  012703  001533                        MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6642  025166  112723  000004                        MOVB    #RMBA,(R3)+
```

```
 6643  025172  112723  000002                  MOVB    #RMWC,(R3)+
 6644  025176  112723  000032                  MOVB    #RMOF,(R3)+
 6645  025202  112723  000006                  MOVB    #RMDA,(R3)+
 6646  025206  112723  000034                  MOVB    #RMDC,(R3)+
 6647  025212  112723  000000                  MOVB    #RMCS1,(R3)+
 6648  025216  112713  000200                  MOVB    #200,(R3)
 6649  025222  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 6650  025226  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
 6651  025230  000240                          NOP                     ;RETURN HERE IF ERROR
 6652  025232  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 6653  025234  000137  025446                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
 6654  025240  004737  044656          50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
 6655  025244  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 6656  025250  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
 6657  025252  000240                          NOP                     ;RETURN HERE IF ERROR
 6658  025254  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 6659  025256  000137  025446                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
 6660  025262                          60$:
 6661  025262  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 6662  025266  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
 6663  025270  000240                          NOP                     ;RETURN HERE IF ERROR
 6664  025272  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 6665  025274  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6666  025276  000137  025446                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
 6667  025302  016237  066770  001140  70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
 6668  025310  042737  157777  001140          BIC     #^tOPI,$GDDAT
 6669  025316  013737  001342  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
 6670  025324  042737  157777  001142          BIC     #^tOPI,$BDDAT
 6671  025332  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS OPI STATUS CORRECT?
 6672  025340  001402                          BEQ     80$             ;YES!!
 6673  025342  104221                          ERROR   221             ;REPORT INCORRECT OPI STATUS
 6674  025344  000440                          BR      110$
 6675  025346  016237  066770  001140  80$:    MOV     FNCDTB(R2),$GDDAT
 6676  025354  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
 6677  025362  001403                          BEQ     90$             ;NO!!
 6678  025364  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
 6679  025372  042737  077777  001140  90$:    BIC     #^tCATA,$GDDAT
 6680  025400  013737  001340  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
 6681  025406  042737  077777  001142          BIC     #^tCATA,$BDDAT
 6682  025414  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
 6683  025422  001402                          BEQ     100$            ;YES!!
 6684  025424  104255                          ERROR   255             ;INCORRECT ATA STATUS
 6685  025426  000407                          BR      110$
 6686  025430  062702  000002          100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
 6687  025434  022702  000076                  CMP     #ILF76,R2       ;DONE??
 6688  025440  103402                          BLO     110$            ;YES !!
 6689  025442  000137  024700                  JMP     10$             ;TEST NEXT FUNCTION
 6690  025446                          110$:
 6691                                  ;;*****************************************************************
 6692                                  ;*TEST 40       ERROR ABORT TESTS
 6693                                  ;;*****************************************************************
 6694  025446                          TST40:
 6695  025446  000004                          SCOPE                   ;SCOPE CALL
 6696  025450  000240                          NOP                     ;START OF TEST
 6697  025452  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 6698  025456  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
```

```
6699  025462  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6700  025466  012737  000040  001226          MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6701
6702  025474  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6703  025500  054130                          .WORD   054130  ;TASK DESCRIPTOR
6704  025502  000404                          BR      10$                     ;GO TO 10$ IF NO ERROR
6705  025504  000240                          NOP                     ;RETURN HERE IF ERROR
6706  025506  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6707  025510  000137  026004                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6708  025514                          10$:
6709  025514  004737  043762                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6710  025520  012702  000000                  MOV     #NOP,R2         ;R2 = FUNCTION CODE
6711  025524                          20$:
6712  025524  004737  053210                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
6713  025530  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6714  025532  000240                          NOP                     ;RETURN HERE IF ERROR
6715  025534  104000                          ERROR                   ;ERROR # DEFINED BY CNTCLR SUBROUTINE
6716  025536  000137  026004                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6717  025542                          30$:
6718  025542  112737  000014  001533          MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
6719  025550  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6720  025556  012737  040000  001412          MOV     #$UNS,RMER10    ;RMER1 OUTPUT BUFFER = #UNS
6721  025564  004737  044316                  JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
6722  025570  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
6723  025572  000240                          NOP                     ;RETURN HERE TO REPORT ERROR
6724  025574  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6725  025576  000137  026004                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
6726  025602                          40$:
6727  025602  012737  000001  001376          MOV     #GO,RMCS10
6728  025610  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6729  025614  012737  105642  001402          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6730  025622  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6731  025630  012737  000000  001404          MOV     #0,RMDA0        ;CLEAR DISK ADDRESS
6732  025636  012737  000000  001432          MOV     #0,RMDC0        ;CLEAR CYLINDER ADDRESS
6733  025644  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
6734  025652  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6735  025656  112723  000004                  MOVB    #RMBA,(R3)+
6736  025662  112723  000002                  MOVB    #RMWC,(R3)+
6737  025666  112723  000032                  MOVB    #RMOF,(R3)+
6738  025672  112723  000006                  MOVB    #RMDA,(R3)+
6739  025676  112723  000034                  MOVB    #RMDC,(R3)+
6740  025702  112723  000000                  MOVB    #RMCS1,(R3)+
6741  025706  112713  000200                  MOVB    #200,(R3)
6742  025712  004737  044316                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6743  025716  000404                          BR      45$                     ;GO TO 45$ IF NO ERROR
6744  025720  000240                          NOP                     ;RETURN HERE IF ERROR
6745  025722  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6746  025724  000137  026004                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6747  025730                          45$:
6748  025730  004737  044656                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6749  025734  004737  044046                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6750  025740  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
6751  025742  000240                          NOP                     ;RETURN HERE IF ERROR
6752  025744  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6753  025746  000137  026004                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6754  025752                          50$:
```

```
                        T40     ERROR ABORT TESTS

6755  025752  004737  045042            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6756  025756  000405                    BR      60$                     ;GO TO 60$ IF NO ERROR
6757  025760  000240                    NOP                     ;RETURN HERE IF ERROR
6758  025762  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6759  025764  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6760  025766  000137  026004            JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6761  025772                    60$:
6762  025772                    70$:
6763  025772  062702  000002            ADD     #2,R2           ;ADVANCE FUNCTION CODE
6764  025776  022702  000076            CMP     #ILF76,R2       ;DONE ALL COMMANDS
6765  026002  103250                    BHIS    20$             ;NO !!
6766  026004                    80$:
6767                            ;;*****************************************************************
6768                            ;*TEST 41        RMR TEST
6769                            ;*****************************************************************
6770  026004                    TST41:
6771  026004  000004                    SCOPE                   ;SCOPE CALL
6772  026006  000240                    NOP                     ;START OF TEST
6773  026010  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6774  026014  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6775  026020  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6776  026024  012737  000041  001226    MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6777
6778  026032  004737  043046            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6779  026036  040000                    .WORD   040000  ;TASK DESCRIPTOR
6780  026040  000404                    BR      10$                     ;GO TO 10$ IF NO ERROR
6781  026042  000240                    NOP                     ;RETURN HERE IF ERROR
6782  026044  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6783  026046  000137  026444            JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6784  026052                    10$:
6785  026052  004737  043762            JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6786  026056                    20$:
6787  026056  112737  000024  001533    MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6788  026064  112737  000200  001534    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6789  026072  012737  000001  001422    MOV     #DMD,RMMR10     ;RMMR1=DMD
6790  026100  004737  044316            JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6791  026104  000404                    BR      30$             ;GO TO 30$ IF NO ERROR
6792  026106  000240                    NOP                     ;RETURN HERE IF ERROR
6793  026110  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6794  026112  000137  026444            JMP     100$            ;GO TO 100$ IF ERROR WAS FOUND
6795  026116                    30$:
6796  026116  112737  000024  001533    MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6797  026124  112737  000200  001534    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6798  026132  012737  001401  001422    MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6799  026140  004737  044316            JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6800  026144  000404                    BR      40$             ;GO TO 40$ IF NO ERROR
6801  026146  000240                    NOP                     ;RETURN HERE IF ERROR
6802  026150  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6803  026152  000137  026444            JMP     100$            ;GO TO 100$ IF ERROR WAS FOUND
6804  026156                    40$:
6805  026156  112737  000000  001533    MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6806  026164  112737  000200  001534    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6807  026172  012737  000023  001376    MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6808  026200  004737  044316            JSR     PC,PUT          ;CALL PUT SUBROUTINE
6809  026204  000404                    BR      50$     ;GO TO 50$ IF NO ERROR
6810  026206  000240                    NOP                     ;RETURN HERE TO REPORT AN ERROR
```

```
6811  026210  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6812  026212  000137  026444                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6813  026216                          50$:
6814  026216  004737  044046                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6815  026222  000404                          BR      60$                     ;GO TO 60$ IF NO ERROR
6816  026224  000240                          NOP                     ;RETURN HERE IF ERROR
6817  026226  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6818  026230  000137  026444                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6819  026234                          60$:
6820  026234  000240                          NOP
6821  026236  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6822  026244  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6823  026252  012737  041401  001422          MOV     #DMD!MUR!MOC!DBEN,RMMR10        ;RMMR1=DMD!MUR!MOC!DBEN
6824  026260  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6825  026264  000404                          BR      70$             ;GO TO 70$ IF NO ERROR
6826  026266  000240                          NOP                     ;RETURN HERE IF ERROR
6827  026270  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6828  026272  000137  026444                  JMP     100$            ;GO TO 100$ IF ERROR WAS FOUND
6829  026276                          70$:
6830  026276  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6831  026304  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6832  026312  012737  000001  001376          MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
6833  026320  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
6834  026324  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6835  026326  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
6836  026330  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6837  026332  000137  026444                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6838  026336                          80$:
6839  026336  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6840  026344  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6841  026352  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
6842  026360  004737  044316                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
6843  026364  000404                          BR      85$     ;GO TO 85$ IF NO ERROR
6844  026366  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
6845  026370  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6846  026372  000137  026444                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6847  026376                          85$:
6848  026376  004737  044046                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6849  026402  000404                          BR      90$                     ;GO TO 90$ IF NO ERROR
6850  026404  000240                          NOP                     ;RETURN HERE IF ERROR
6851  026406  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6852  026410  000137  026444                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6853  026414                          90$:
6854  026414  000240                          NOP
6855  026416  032737  000004  001342          BIT     #RMR,RMER1I             ;IS RMR SET ??
6856  026424  001007                          BNE     100$                    ;YES !!
6857  026426  012737  000004  001140          MOV     #RMR,$GDDAT             ;EXPECTED STATUS
6858  026434  013737  001342  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6859  026442  104222                          ERROR   222                     ;RMR DID NOT SET
6860  026444                          100$:
6861                                  ;;*****************************************************************
6862                                  ;*TEST 42       PARITY ERROR TEST
6863                                  ;;*****************************************************************
6864  026444                          TST42:
6865  026444  000004                          SCOPE                   ;SCOPE CALL
6866  026446  000240                          NOP                     ;START OF TEST
```

```
6867  026450  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6868  026454  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6869  026460  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6870  026464  012737  000042  001226          MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6871
6872  026472  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6873  026476  040000                          .WORD   040000  ;TASK DESCRIPTOR
6874  026500  000404                          BR      10$                     ;GO TO 10$ IF NO ERROR
6875  026502  000240                          NOP                     ;RETURN HERE IF ERROR
6876  026504  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6877  026506  000137  026670                  JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
6878  026512                          10$:
6879  026512  012702  000001                  MOV     #1,R2                   ;R2 = DATA PATTERN
6880  026516                          20$:
6881  026516  004737  053210                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
6882  026522  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6883  026524  000240                          NOP                     ;RETURN HERE IF ERROR
6884  026526  104000                          ERROR                   ;ERROR # DEFINED BY CNTCLR SUBROUTINE
6885  026530  000137  026670                  JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
6886  026534                          30$:
6887  026534  111103                          MOVB    (R1),R3                 ;SETUP RMCS2
6888  026536  042703  177770                  BIC     #^CUNTMSK,R3
6889  026542  052703  000020                  BIS     #PAT,R3
6890  026546  010337  001406                  MOV     R3,RMCS20               ;OUTPUT VALUE TO RMCS2
6891  026552  010237  001404                  MOV     R2,RMDA0                ;VALUE TO RMDA
6892  026556  012703  001533                  MOV     #PUTINX,R3              ;WRITE REGISTER OUTPUT INDEX
6893  026562  112723  000010                  MOVB    #RMCS2,(R3)+
6894  026566  112723  000006                  MOVB    #RMDA,(R3)+
6895  026572  112723  000200                  MOVB    #200,(R3)+
6896
6897  026576  004737  044316                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6898  026602  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
6899  026604  000240                          NOP                     ;RETURN HERE IF ERROR
6900  026606  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6901  026610  000137  026670                  JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
6902  026614                          40$:
6903  026614  004737  044046                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6904  026620  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
6905  026622  000240                          NOP                     ;RETURN HERE IF ERROR
6906  026624  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6907  026626  000137  026670                  JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
6908  026632                          50$:
6909  026632  032737  000010  001342          BIT     #PAR,RMER1I             ;IS PARITY ERROR SET ??
6910  026640  001011                          BNE     60$                     ;YES !!
6911  026642  012737  000010  001140          MOV     #PAR,$GDDAT             ;EXPECTED STATUS
6912  026650  013737  001342  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6913  026656  010237  001174                  MOV     R2,$TMP0                ;DATA PATTERN
6914  026662  104223                          ERROR   223                     ;PAR DID NOT SET
6915  026664                          60$:
6916  026664  006302                          ASL     R2                      ;ADVANCE DATA PATTERN
6917  026666  001313                          BNE     20$                     ;BRANCH IF NOT DONE
6918  026670                          70$:
6919                                  ;;*****************************************************************
6920                                  ;*TEST 43       ILLEGAL REGISTER TEST
6921                                  ;;*****************************************************************
6922  026670                          TST43:
```

```
6923  026670  000004                        SCOPE                   ;SCOPE CALL
6924  026672  000240                        NOP                     ;START OF TEST
6925  026674  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6926  026700  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6927  026704  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6928  026710  012737  000043  001226        MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6929
6930  026716  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6931  026722  040000                        .WORD   040000  ;TASK DESCRIPTOR
6932  026724  000404                        BR      10$                     ;GO TO 10$ IF NO ERROR
6933  026726  000240                        NOP                     ;RETURN HERE IF ERROR
6934  026730  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6935  026732  000137  027222                JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
6936  026736  005005              10$:      CLR     R5              ;R5 = EXPECTED STATUS
6937  026740  004737  043762                JSR     PC,GETSTS       ;SETUP FOR STATUS
6938  026744  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
6939  026750  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
6940  026754  012737  027220  000004        MOV     #110$,ERRVEC    ;SETUP FOR BUS TIMEOUT
6941  026762  012737  000300  000006        MOV     #PR6,ERRVEC+2
6942  026770  005002                        CLR     R2              ;R2=REGISTER INDEX
6943  026772                        20$:
6944  026772  004737  053210                JSR     PC,CNTCLR
6945  026776  000404                        BR      30$     ;GO TO 30$ IF NO ERROR
6946  027000  000240                        NOP                     ;RETURN HERE IF ERROR
6947  027002  104000                        ERROR                   ;ERROR NUMBER DEFINED BY SUB
6948  027004  000137  027130                JMP     70$     ;GO TO 70$ IF ERROR WAS FOUND
6949  027010  010003              30$:      MOV     R0,R3           ;R3=REGISTER ADDRESS
6950  027012  060203                        ADD     R2,R3
6951  027014  005013                        CLR     (R3)            ;CLEAR THE REGISTER
6952  027016  004737  051536                JSR     PC,DEVSEL       ;GO SELECT DEVICE
6953  027022  000404                        BR      35$                     ;GO TO 35$ IF NO ERROR
6954  027024  000240                        NOP                     ;RETURN HERE IF ERROR
6955  027026  104000                        ERROR                   ;ERROR # DEFINED BY DEVSEL SUBROUTINE
6956  027030  000137  027222                JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
6957  027034                        35$:
6958  027034  004737  044046                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6959  027040  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
6960  027042  000240                        NOP                     ;RETURN HERE IF ERROR
6961  027044  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
6962  027046  000137  027130                JMP     70$     ;GO TO 70$ IF ERROR WAS FOUND
6963  027052                        40$:
6964  027052  004737  045042                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6965  027056  000405                        BR      50$                     ;GO TO 50$ IF NO ERROR
6966  027060  000240                        NOP                     ;RETURN HERE IF ERROR
6967  027062  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6968  027064  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6969  027066  000137  027130                JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
6970  027072  010537  001140        50$:    MOV     R5,$GDDAT
6971  027076  013737  001342  001142        MOV     RMER2I,$BDDAT   ;GET DRIVE'S ILR STATUS
6972  027104  042737  177775  001142        BIC     #^C2ILR,$BDDAT
6973  027112  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS ILR STATUS OK??
6974  027120  001403                        BEQ     70$             ;YES!!
6975  027122  010237  001174                MOV     R2,$TMP0        ;SAVE R2 FOR ERROR DATA
6976  027126  104256                        ERROR   256
6977  027130  062702  000002        70$:    ADD     #2,R2
6978  027134  022702  000050                CMP     #RMBAE,R2       ;IS NEXT REGISTER RMBAE??
```

```
 6979  027140  001015                          BNE     80$                     ;NO!!
 6980  027142  012760  001400  000000          MOV     #A16!A17,RMCS1(R0)      ;SET A16 & A17
 6981  027150  016003  000050                  MOV     RMBAE(R0),R3    ;IS THIS AN RH70??
 6982  027154  042703  177774                  BIC     #↑C<BIT0!BIT1>,R3       ;CLEAR ALL BUT ADDRESS BITS
 6983  027160  022703  000003                  CMP     #BIT0!BIT1,R3   ;ARE ADDRESS BITS ON ??
 6984  027164  001414                          BEQ     100$            ;YES
 6985  027166  012705  000002                  MOV     #ILR,R5         ;NO - EXPECT ILR=1
 6986  027172  000411                          BR      100$
 6987  027174  022702  000054          80$:    CMP     #RMCS3+2,R2     ;SHOULD ILR BE SET??
 6988  027200  001003                          BNE     90$             ;NO!!
 6989  027202  012705  000002                  MOV     #ILR,R5         ;YES!!
 6990  027206  000403                          BR      100$
 6991  027210  022702  000076          90$:    CMP     #76,R2          ;DONE??
 6992  027214  103402                          BLO     120$            ;YES!!
 6993  027216  000665                  100$:   BR      20$
 6994
 6995  027220  022626                  110$:   CMP     (SP)+,(SP)+
 6996  027222                          120$:
 6997  027222  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 6998  027226  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 6999                                  ;;**************************************************************
 7000                                  ;*TEST 44        SEEK LAST CYLINDER
 7001
 7002                                  ;;**************************************************************
 7003  027232                          TST44:
 7004  027232  000004                          SCOPE                   ;SCOPE CALL
 7005  027234  000240                          NOP                     ;START OF TEST
 7006  027236  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
 7007  027244  012737  027260  001122          MOV     #1$,$LPADR
 7008  027252  012737  027260  001124          MOV     #1$,$LPERR
 7009  027260                          1$:
 7010  027260  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 7011  027264  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 7012  027270  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 7013  027274  012737  000044  001226          MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 7014  027302                          10$:
 7015  027302  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7016  027306  054130                          .WORD   054130  ;TASK DESCRIPTOR
 7017  027310  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
 7018  027312  000240                          NOP                     ;RETURN HERE IF ERROR
 7019  027314  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7020  027316  000137  027534                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
 7021  027322                          80$:
 7022  027322  012737  000000  001404          MOV     #0,RMDA0        ;TRACK = SECTOR = 0
 7023  027330  012737  001466  001432          MOV     #822.,RMDC0     ;CYLINDER = 822
 7024  027336  012737  000005  001376          MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
 7025  027344  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
 7026  027350  112722  000034                  MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
 7027  027354  112722  000006                  MOVB    #RMDA,(R2)+
 7028  027360  112722  000000                  MOVB    #RMCS1,(R2)+
 7029  027364  112722  000200                  MOVB    #200,(R2)+      ;WRITE TERMINATOR
 7030  027370  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7031  027374  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
 7032  027376  000240                          NOP                     ;RETURN HERE IF ERROR
 7033  027400  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 7034  027402  000137  027534                  JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
```

```
7035  027406  004737  043762          90$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7036  027412  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
7037  027416  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7038  027422  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
7039  027424  000240                          NOP                     ;RETURN HERE IF ERROR
7040  027426  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7041  027430  000137  027534                  JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7042  027434                          100$:
7043  027434  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7044  027440  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7045  027442  000240                          NOP                     ;RETURN HERE IF ERROR
7046  027444  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7047  027446  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7048  027450  000137  027534                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7049  027454                          110$:
7050  027454  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7051  027460  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7052  027462  000240                          NOP                     ;RETURN HERE IF ERROR
7053  027464  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7054  027466  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7055  027470  000137  027534                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7056  027474                          120$:
7057  027474  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7058  027500  115760                          .WORD   NOTMSK          ;MASK FOR RMER1
7059  027502  000010                          .WORD   DPE             ;MASK FOR RMER2
7060  027504  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
7061  027506  000240                          NOP                     ;RETURN HERE IF ERROR
7062  027510  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7063  027512  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7064  027514  000137  027534                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7065  027520                          130$:
7066  027520  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7067  027524  000403                          BR      140$                    ;GO TO 140$ IF NO ERROR
7068  027526  000240                          NOP                     ;RETURN HERE IF ERROR
7069  027530  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7070  027532  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7071  027534                          140$:
7072                                  ;;*****************************************************************
7073                                  ;*TEST 45        SEEK FIRST CYLINDER
7074
7075                                  ;;*****************************************************************
7076  027534                          TST45:
7077  027534  000004                          SCOPE                   ;SCOPE CALL
7078  027536  000240                          NOP                     ;START OF TEST
7079  027540  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7080  027546  012737  027562  001122          MOV     #1$,$LPADR
7081  027554  012737  027562  001124          MOV     #1$,$LPERR
7082  027562                          1$:
7083  027562  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7084  027566  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7085  027572  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7086  027576  012737  000045  001226          MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7087  027604                          10$:
7088  027604  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7089  027610  054130                          .WORD   054130  ;TASK DESCRIPTOR
7090  027612  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
```

```
7091  027614  000240                        NOP                     ;RETURN HERE IF ERROR
7092  027616  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7093  027620  000137  030036                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7094  027624                        80$:
7095  027624  012737  000000  001404        MOV     #0,RMDAO        ;TRACK = SECTOR = 0
7096  027632  012737  000000  001432        MOV     #0,RMDCO        ;CYLINDER = 0
7097  027640  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
7098  027646  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7099  027652  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7100  027656  112722  000006                MOVB    #RMDA,(R2)+
7101  027662  112722  000000                MOVB    #RMCS1,(R2)+
7102  027666  112722  000200                MOVB    #200,(R2)+      ;WRITE TERMINATOR
7103  027672  004737  044316                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7104  027676  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7105  027700  000240                        NOP                     ;RETURN HERE IF ERROR
7106  027702  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7107  027704  000137  030036                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7108  027710  004737  043762        90$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7109  027714  004737  044656                JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
7110  027720  004737  044046                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7111  027724  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
7112  027726  000240                        NOP                     ;RETURN HERE IF ERROR
7113  027730  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7114  027732  000137  030036                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7115  027736                        100$:
7116  027736  004737  045042                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7117  027742  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
7118  027744  000240                        NOP                     ;RETURN HERE IF ERROR
7119  027746  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7120  027750  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7121  027752  000137  030036                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7122  027756                        110$:
7123  027756  004737  051750                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7124  027762  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
7125  027764  000240                        NOP                     ;RETURN HERE IF ERROR
7126  027766  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7127  027770  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7128  027772  000137  030036                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7129  027776                        120$:
7130  027776  004737  061416                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7131  030002  115760                        .WORD   NDTMSK          ;MASK FOR RMER1
7132  030004  000010                        .WORD   DPE             ;MASK FOR RMER2
7133  030006  000405                        BR      130$                    ;GO TO 130$ IF NO ERROR
7134  030010  000240                        NOP                     ;RETURN HERE IF ERROR
7135  030012  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7136  030014  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7137  030016  000137  030036                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7138  030022                        130$:
7139  030022  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7140  030026  000403                        BR      140$                    ;GO TO 140$ IF NO ERROR
7141  030030  000240                        NOP                     ;RETURN HERE IF ERROR
7142  030032  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7143  030034  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7144  030036                        140$:
7145                                ;;*****************************************************************
7146                                ;*TEST 46        SEEK PRIME CYLINDERS
```

```
7147
7148                                    ;;*****************************************************************
7149  030036                            T$T46:
7150  030036  000004                            SCOPE                   ;SCOPE CALL
7151  030040  000240                            NOP                     ;START OF TEST
7152  030042  012737  000001  001206            MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7153  030050  012737  030064  001122            MOV     #1$,$LPADR
7154  030056  012737  030064  001124            MOV     #1$,$LPERR
7155  030064                            1$:
7156  030064  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7157  030070  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7158  030074  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7159  030100  012737  000046  001226            MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7160  030106  012737  000001  001432            MOV     #1,RMDC0        ;FIRST CYLINDER WILL BE 1
7161  030114                            10$:
7162  030114  004737  043046                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7163  030120  054130                            .WORD   054130  ;TASK DESCRIPTOR
7164  030122  000404                            BR      80$                     ;GO TO 80$ IF NO ERROR
7165  030124  000240                            NOP                     ;RETURN HERE IF ERROR
7166  030126  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7167  030130  000137  030364                    JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
7168  030134  012737  000000  001404  80$:      MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7169  030142  012737  000005  001376            MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
7170  030150  012702  001533                    MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7171  030154  112722  000034                    MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7172  030160  112722  000006                    MOVB    #RMDA,(R2)+
7173  030164  112722  000000                    MOVB    #RMCS1,(R2)+
7174  030170  112722  000200                    MOVB    #200,(R2)+      ;WRITE TERMINATOR
7175  030174  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7176  030200  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
7177  030202  000240                            NOP                     ;RETURN HERE IF ERROR
7178  030204  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
7179  030206  000137  030340                    JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7180  030212  004737  043762  90$:      JSR     PC,GETST$       ;SETUP FOR STATUS FETCH
7181  030216  004737  044656                    JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
7182  030222  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7183  030226  000404                            BR      100$    ;GO TO 100$ IF NO ERROR
7184  030230  000240                            NOP                     ;RETURN HERE IF ERROR
7185  030232  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
7186  030234  000137  030340                    JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7187  030240                            100$:
7188  030240  004737  045042                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7189  030244  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
7190  030246  000240                            NOP                     ;RETURN HERE IF ERROR
7191  030250  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7192  030252  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7193  030254  000137  030340                    JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7194  030260                            110$:
7195  030260  004737  051750                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7196  030264  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
7197  030266  000240                            NOP                     ;RETURN HERE IF ERROR
7198  030270  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7199  030272  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7200  030274  000137  030340                    JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7201  030300                            120$:
7202  030300  004737  061416                    JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
```

```
7203  030304  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
7204  030306  000010                          .WORD   DPE             ;MASK FOR RMER2
7205  030310  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
7206  030312  000240                          NOP                     ;RETURN HERE IF ERROR
7207  030314  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7208  030316  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7209  030320  000137  030340                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7210  030324                          130$:
7211  030324  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7212  030330  000403                          BR      140$                    ;GO TO 140$ IF NO ERROR
7213  030332  000240                          NOP                     ;RETURN HERE IF ERROR
7214  030334  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7215  030336  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7216  030340  013746  001432          140$:   MOV     RMDC0,-(SP)     ;SHIFT TO NEXT PRIME CYLINDER
7217  030344  006316                          ASL     (SP)
7218  030346  011637  001432                  MOV     (SP),RMDC0
7219  030352  022726  001000                  CMP     #512.,(SP)+     ;IS THE TEST DONE??
7220  030356  103402                          BLO     150$            ;YES!!
7221  030360  000137  030114                  JMP     10$             ;GO DO NEXT CYLINDER
7222  030364                          150$:
7223
7224
7225                                  ;;*****************************************************************
7226                                  ;*TEST 47        SEEK ZERO DIFFERENCE
7227
7228                                  ;;*****************************************************************
7229  030364                          TST47:
7230  030364  000004                          SCOPE                   ;SCOPE CALL
7231  030366  000240                          NOP                     ;START OF TEST
7232  030370  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7233  030376  012737  030412  001122          MOV     #1$,$LPADR
7234  030404  012737  030412  001124          MOV     #1$,$LPERR
7235  030412                          1$:
7236  030412  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7237  030416  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7238  030422  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7239  030426  012737  000047  001226          MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7240  030434                          10$:
7241  030434  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7242  030440  054130                          .WORD   054130  ;TASK DESCRIPTOR
7243  030442  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
7244  030444  000240                          NOP                     ;RETURN HERE IF ERROR
7245  030446  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7246  030450  000137  030706                  JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7247  030454                          80$:
7248  030454  012703  000001                  MOV     #1,R3           ;R3 = NUMBER OF SEEKS
7249  030460  012737  000000  001432          MOV     #0,RMDC0        ;CYLINDER = 0
7250  030466  012737  000000  001404          MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7251  030474  012737  000005  001376          MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
7252  030502  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
7253  030506  112722  000006                  MOVB    #RMDA,(R2)+     ;WRITE REGISTER INDEX TABLE
7254  030512  112722  000034                  MOVB    #RMDC,(R2)+
7255  030516  112722  000000                  MOVB    #RMCS1,(R2)+
7256  030522  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
7257  030526                          90$:
7258  030526  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
```

```
7259  030532  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
7260  030534  000240                          NOP                     ;RETURN HERE IF ERROR
7261  030536  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7262  030540  000137  030706                  JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7263  030544  004737  043762          100$:   JSR     PC,GETSTS       ;SETUP FOR READING STATUS
7264  030550  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
7265  030554  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7266  030560  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
7267  030562  000240                          NOP                     ;RETURN HERE IF ERROR
7268  030564  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7269  030566  000137  030706                  JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7270  030572                          120$:
7271  030572  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7272  030576  000405                          BR      130$            ;GO TO 130$ IF NO ERROR
7273  030600  000240                          NOP                     ;RETURN HERE IF ERROR
7274  030602  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7275  030604  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7276  030606  000137  030706                  JMP     170$            ;GO TO 170$ IF ERROR WAS FOUND
7277  030612                          130$:
7278  030612  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7279  030616  000405                          BR      140$            ;GO TO 140$ IF NO ERROR
7280  030620  000240                          NOP                     ;RETURN HERE IF ERROR
7281  030622  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7282  030624  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7283  030626  000137  030706                  JMP     170$            ;GO TO 170$ IF ERROR WAS FOUND
7284  030632                          140$:
7285  030632  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7286  030636  115760                          .WORD   ND!MSK          ;MASK FOR RMER1
7287  030640  000010                          .WORD   DPE             ;MASK FOR RMER2
7288  030642  000405                          BR      150$            ;GO TO 150$ IF NO ERROR
7289  030644  000240                          NOP                     ;RETURN HERE IF ERROR
7290  030646  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7291  030650  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7292  030652  000137  030706                  JMP     170$            ;GO TO 170$ IF ERROR WAS FOUND
7293  030656                          150$:
7294  030656  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7295  030662  000405                          BR      160$            ;GO TO 160$ IF NO ERROR
7296  030664  000240                          NOP                     ;RETURN HERE IF ERROR
7297  030666  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7298  030670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7299  030672  000137  030706                  JMP     170$            ;GO TO 170$ IF ERROR WAS FOUND
7300  030676                          160$:
7301  030676  005303                          DEC     R3              ;DONE ALL SEEKS??
7302  030700  100402                          BMI     170$            ;YES!!
7303  030702  000137  030526                  JMP     90$             ;NO - GO DO NEXT SEEK
7304  030706                          170$:
7305                                  ;;*****************************************************************
7306                                  ;*TEST 50        SEEK MAXIMUM DIFFERENCE FORWARD
7307
7308                                  ;;*****************************************************************
7309  030706                          T$T50:
7310  030706  000004                          SCOPE                   ;SCOPE CALL
7311  030710  000240                          NOP                     ;START OF TEST
7312  030712  012737  000002  001206          MOV     #2,$TIMES       ;LOAD ITERATION COUNT
7313  030720  012737  030734  001122          MOV     #1$,$LPADR
7314  030726  012737  030734  001124          MOV     #1$,$LPERR
```

```
 7315  030734
 7316  030734  012706  001100                1$:
 7317  030740  013700  001276                      MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 7318  030744  013701  001446                      MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 7319  030750  012737  000050  001226              MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 7320  030756                                      MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 7321  030756  004737  043046                10$:
 7322  030762  054130                            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7323  030764  000404                            .WORD   054130  ;TASK DESCRIPTOR
 7324  030766  000240                            BR      80$                     ;GO TO 80$ IF NO ERROR
 7325  030770  104000                            NOP                     ;RETURN HERE IF ERROR
 7326  030772  000137  031234                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7327  030776                                    JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 7328  030776  012737  000000  001432        80$:
 7329  031004  012737  000000  001404              MOV     #0,RMDC0                ;SEEK TO CYLINDER 0
 7330  031012  012737  000005  001376              MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 7331  031020  012737  001533                      MOV     #SEEK!GO,RMCS10         ;FUNCTION CODE FOR SEEK
 7332  031024  112737  000006                      MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER TABLE
 7333  031030  112722  000034                      MOVB    #RMDA,(R2)+             ;WRITE REGISTER INDEX TABLE
 7334  031034  112722  000000                      MOVB    #RMDC,(R2)+
 7335  031040  112722  000200                      MOVB    #RMCS1,(R2)+
 7336  031044                                      MOVB    #200,(R2)+              ;TERMINATE THE TABLE
 7337  031044  004737  044316                90$:
 7338  031050  000404                            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7339  031052  000240                            BR      100$    ;GO TO 100$ IF NO ERROR
 7340  031054  104000                            NOP                     ;RETURN HERE IF ERROR
 7341  031056  000137  031234                    ERROR                   ;ERROR DEFINED BY PUT SUB
 7342  031062                                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 7343  031062  004737  043762                100$:
 7344  031066  004737  044656                      JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
 7345  031072  004737  044046                      JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
 7346  031076  000404                            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7347  031100  000240                            BR      110$    ;GO TO 110$ IF NO ERROR
 7348  031102  104000                            NOP                     ;RETURN HERE IF ERROR
 7349  031104  000137  031234                    ERROR                   ;ERROR DEFINED BY GET SUB
 7350  031110                                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 7351  031110  004737  045042                110$:
 7352  031114  000405                            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7353  031116  000240                            BR      120$                    ;GO TO 120$ IF NO ERROR
 7354  031120  104000                            NOP                     ;RETURN HERE IF ERROR
 7355  031122  004736                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7356  031124  000137  031234                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7357  031130                                    JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 7358  031130  004737  051750                120$:
 7359  031134  000405                            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 7360  031136  000240                            BR      130$                    ;GO TO 130$ IF NO ERROR
 7361  031140  104000                            NOP                     ;RETURN HERE IF ERROR
 7362  031142  004736                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 7363  031144  000137  031234                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7364  031150                                    JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 7365  031150  004737  061416                130$:
 7366  031154  115760                            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 7367  031156  000010                            .WORD   NOTMSK          ;MASK FOR RMER1
 7368  031160  000405                            .WORD   DPE             ;MASK FOR RMER2
 7369  031162  000240                            BR      140$                    ;GO TO 140$ IF NO ERROR
 7370  031164  104000                            NOP                     ;RETURN HERE IF ERROR
                                                 ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
```

```
7371  031166  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7372  031170  000137  031234                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7373  031174                          140$:
7374  031174  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7375  031200  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7376  031202  000240                        NOP                     ;RETURN HERE IF ERROR
7377  031204  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7378  031206  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7379  031210  000137  031234                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7380  031214                          150$:
7381  031214  005737  001432                TST     RMDC0           ;IS TEST DONE??>
7382  031220  001005                        BNE     160$            ;YES!!
7383  031222  012737  001466  001432        MOV     #822.,RMDC0     ;NO - SEEK TO LAST CYLINDER
7384  031230  000137  031044                JMP     90$
7385  031234                          160$:
7386                                  ;;**********************************************************************
7387                                  ;*TEST 51        SEEK ADJACENT FORWARD
7388
7389                                  ;;**********************************************************************
7390  031234                          TST51:
7391  031234  000004                        SCOPE                   ;SCOPE CALL
7392  031236  000240                        NOP                     ;START OF TEST
7393  031240  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7394  031246  012737  031262  001122        MOV     #1$,$LPADR
7395  031254  012737  031262  001124        MOV     #1$,$LPERR
7396  031262                          1$:
7397  031262  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7398  031266  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7399  031272  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7400  031276  012737  000051  001226        MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7401  031304                          10$:
7402  031304  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7403  031310  054130                        .WORD   054130  ;TASK DESCRIPTOR
7404  031312  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
7405  031314  000240                        NOP                     ;RETURN HERE IF ERROR
7406  031316  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7407  031320  000137  031562                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7408  031324                          80$:
7409  031324  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER = 0
7410  031332  012737  000000  001404        MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7411  031340  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
7412  031346  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
7413  031352  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7414  031356  112722  000006                MOVB    #RMDA,(R2)+
7415  031362  112722  000000                MOVB    #RMCS1,(R2)+
7416  031366  112722  000200                MOVB    #200,(R2)+      ;TERMINATE REGISTER TABLE
7417  031372                          90$:
7418  031372  004737  044316                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7419  031376  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
7420  031400  000240                        NOP                     ;RETURN HERE IF ERROR
7421  031402  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7422  031404  000137  031562                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7423  031410  004737  043762        100$:   JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7424  031414  004737  044656                JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
7425  031420  004737  044046                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7426  031424  000404                        BR      110$    ;GO TO 110$ IF NO ERROR
```

```
7427  031426  000240                          NOP                     ;RETURN HERE IF ERROR
7428  031430  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7429  031432  000137  031562                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7430  031436                          110$:
7431  031436  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7432  031442  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7433  031444  000240                          NOP                     ;RETURN HERE IF ERROR
7434  031446  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7435  031450  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7436  031452  000137  031562                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7437  031456                          120$:
7438  031456  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7439  031462  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
7440  031464  000240                          NOP                     ;RETURN HERE IF ERROR
7441  031466  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7442  031470  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7443  031472  000137  031562                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7444  031476                          130$:
7445  031476  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7446  031502  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
7447  031504  000010                          .WORD   DPE             ;MASK FOR RMER2
7448  031506  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
7449  031510  000240                          NOP                     ;RETURN HERE IF ERROR
7450  031512  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7451  031514  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7452  031516  000137  031562                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7453  031522                          140$:
7454  031522  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7455  031526  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7456  031530  000240                          NOP                     ;RETURN HERE IF ERROR
7457  031532  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7458  031534  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7459  031536  000137  031562                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7460  031542                          150$:
7461  031542  005737  001432                  TST     RMDC0           ;FIRST TIME THROUGH??
7462  031546  001005                          BNE     160$            ;NO!!
7463  031550  012737  000001  001432          MOV     #1,RMDC0        ;YES - SEEK TO LAST CYLINDER
7464  031556  000137  031372                  JMP     90$
7465  031562                          160$:
7466                                  ;;*****************************************************************
7467                                  ;*TEST 52        SEEK ADJACENT REVERSE
7468
7469                                  ;;*****************************************************************
7470  031562                          TST52:
7471  031562  000004                          SCOPE                   ;SCOPE CALL
7472  031564  000240                          NOP                     ;START OF TEST
7473  031566  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7474  031574  012737  031610  001122          MOV     #1$,$LPADR
7475  031602  012737  031610  001124          MOV     #1$,$LPERR
7476  031610                          1$:
7477  031610  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7478  031614  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7479  031620  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7480  031624  012737  000052  001226          MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7481  031632                          10$:
7482  031632  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
7483  031636  054130                          .WORD   054130  ;TASK DESCRIPTOR
7484  031640  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
7485  031642  000240                          NOP                     ;RETURN HERE IF ERROR
7486  031644  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7487  031646  000137  032110                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7488  031652                          80$:
7489  031652  012737  001466  001432          MOV     #822.,RMDC0             ;START AT LAST CYLINDER
7490  031660  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7491  031666  012737  000005  001376          MOV     #SEEK!GO,RMCS10         ;FUNCTION CODE FOR SEEK
7492  031674  012702  001533                  MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER INDEX
7493  031700  112722  000006                  MOVB    #RMDA,(R2)+             ;WRITE REGISTER INDEX TABLE
7494  031704  112722  000034                  MOVB    #RMDC,(R2)+
7495  031710  112722  000000                  MOVB    #RMCS1,(R2)+
7496  031714  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
7497  031720                          90$:
7498  031720  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7499  031724  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
7500  031726  000240                          NOP                     ;RETURN HERE IF ERROR
7501  031730  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7502  031732  000137  032110                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7503  031736  004737  043762          100$:   JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
7504  031742  004737  044656                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
7505  031746  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7506  031752  000404                          BR      110$    ;GO TO 110$ IF NO ERROR
7507  031754  000240                          NOP                     ;RETURN HERE IF ERROR
7508  031756  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7509  031760  000137  032110                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7510  031764                          110$:
7511  031764  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7512  031770  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7513  031772  000240                          NOP                     ;RETURN HERE IF ERROR
7514  031774  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7515  031776  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7516  032000  000137  032110                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7517  032004                          120$:
7518  032004  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7519  032010  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
7520  032012  000240                          NOP                     ;RETURN HERE IF ERROR
7521  032014  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7522  032016  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7523  032020  000137  032110                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7524  032024                          130$:
7525  032024  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7526  032030  115760                          .WORD   NOTMSK          ;MASK FOR RMER1
7527  032032  000010                          .WORD   DPE             ;MASK FOR RMER2
7528  032034  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
7529  032036  000240                          NOP                     ;RETURN HERE IF ERROR
7530  032040  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7531  032042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7532  032044  000137  032110                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7533  032050                          140$:
7534  032050  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7535  032054  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7536  032056  000240                          NOP                     ;RETURN HERE IF ERROR
7537  032060  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7538  032062  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
7539  032064  000137  032110                        JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7540  032070                                150$:
7541  032070  022737  001466  001432                CMP     #822.,RMDC0             ;IS THIS THE FIRST SEEK??
7542  032076  001004                                BNE     160$                    ;NO!!
7543  032100  005337  001432                        DEC     RMDC0                   ;YES - SEEK TO ADJACENT CYLINDER
7544  032104  000137  031720                        JMP     90$                     ;GO SEEK
7545  032110                                160$:
7546                                        ;;*****************************************************************
7547                                        ;*TEST 53       SEEK INVALID SECTOR
7548                                        ;;*****************************************************************
7549  032110                                TST53:
7550  032110  000004                               SCOPE                           ;SCOPE CALL
7551  032112  000240                               NOP                             ;START OF TEST
7552  032114  012706  001100                       MOV     #STACK,SP               ;INITIALIZE STACK POINTER
7553  032120  013700  001276                       MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
7554  032124  013701  001446                       MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
7555  032130  012737  000053  001226               MOV     #53,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
7556
7557  032136                                10$:
7558  032136  004737  043046                       JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
7559  032142  054130                               .WORD   054130  ;TASK DESCRIPTOR
7560  032144  000404                               BR      80$                             ;GO TO 80$ IF NO ERROR
7561  032146  000240                               NOP                             ;RETURN HERE IF ERROR
7562  032150  104000                               ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7563  032152  000137  032476                       JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
7564  032156  012737  000005  001376  80$:         MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
7565  032164  012737  000000  001430               MOV     #0,RMOF0                ;RESET FORMAT 16 BIT
7566  032172  012737  000000  001432               MOV     #0,RMDC0                ;CYLINDER=0
7567  032200  012737  000036  001404               MOV     #30.,RMDA0              ;SECTOR=30,TRACK=0
7568  032206  012702  001533                       MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER TABLE
7569  032212  112722  000032                       MOVB    #RMOF,(R2)+             ;WRITE REGISTER INDEX TABLE
7570  032216  112722  000006                       MOVB    #RMDA,(R2)+
7571  032222  112722  000034                       MOVB    #RMDC,(R2)+
7572  032226  112722  000000                       MOVB    #RMCS1,(R2)+
7573  032232  112722  000200                       MOVB    #200,(R2)+              ;TERMINATE TABLE
7574  032236  004737  043762                       JSR     PC,GETSTS               ;SETUP INPUT REGISTER INDEX
7575  032242                                90$:
7576  032242  004737  053210                       JSR     PC,CNTCLR
7577  032246  000404                               BR      95$     ;GO TO 95$ IF NO ERROR
7578  032250  000240                               NOP                             ;RETURN HERE IF ERROR
7579  032252  104000                               ERROR                           ;ERROR NUMBER DEFINED BY SUB
7580  032254  000137  032476                       JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7581  032260                                95$:
7582  032260  004737  044046                       JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
7583  032264  000404                               BR      100$    ;GO TO 100$ IF NO ERROR
7584  032266  000240                               NOP                             ;RETURN HERE IF ERROR
7585  032270  104000                               ERROR                           ;ERROR DEFINED BY GET SUB
7586  032272  000137  032476                       JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7587  032276                                100$:
7588  032276  004737  053326                       JSR     PC,CLRSTS               ;GO VERIFY CONTROLLER CLEAR OPERATION
7589  032302  000405                               BR      110$                            ;GO TO 110$ IF NO ERROR
7590  032304  000240                               NOP                             ;RETURN HERE IF ERROR
7591  032306  104000                               ERROR                           ;ERROR # DEFINED BY CLRSTS SUBROUTINE
7592  032310  004736                               JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7593  032312  000137  032476                       JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
7594  032316                                110$:
```

```
7595  032316  004737  044316          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7596  032322  000404                  BR      120$    ;GO TO 120$ IF NO ERROR
7597  032324  000240                  NOP                     ;RETURN HERE IF ERROR
7598  032326  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7599  032330  000137  032456          JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7600  032334  004737  044656  120$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
7601  032340  004737  044046          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7602  032344  000404                  BR      125$                    ;GO TO 125$ IF NO ERROR
7603  032346  000240                  NOP                     ;RETURN HERE IF ERROR
7604  032350  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7605  032352  000137  032476          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7606  032356                  125$:
7607  032356  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7608  032362  000405                  BR      130$                    ;GO TO 130$ IF NO ERROR
7609  032364  000240                  NOP                     ;RETURN HERE IF ERROR
7610  032366  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7611  032370  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7612  032372  000137  032476          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7613  032376                  130$:
7614  032376  004737  051750          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7615  032402  000405                  BR      140$                    ;GO TO 140$ IF NO ERROR
7616  032404  000240                  NOP                     ;RETURN HERE IF ERROR
7617  032406  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7618  032410  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7619  032412  000137  032476          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7620  032416                  140$:
7621  032416                  150$:
7622  032416  004737  061416          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7623  032422  117760                  .WORD   IAE!NDTMSK              ;MASK FOR RMER1
7624  032424  000010                  .WORD   DPE             ;MASK FOR RMER2
7625  032426  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
7626  032430  000240                  NOP                     ;RETURN HERE IF ERROR
7627  032432  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7628  032434  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7629  032436  000137  032456          JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7630  032442                  160$:
7631  032442  004737  045674          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7632  032446  000403                  BR      170$                    ;GO TO 170$ IF NO ERROR
7633  032450  000240                  NOP                     ;RETURN HERE IF ERROR
7634  032452  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7635  032454  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7636  032456  005237  001404  170$:   INC     RMDAO           ;INCREMENT SECTOR ADDRESS
7637  032462  022737  000037  001404  CMP     #31.,RMDAO      ;DONE??
7638  032470  103402                  BLO     180$            ;YES!!
7639  032472  000137  032242          JMP     90$             ;NO - TEST NEXT SECTOR
7640  032476                  180$:
7641
7642                          ;;*****************************************************************
7643                          ;*TEST 54        SEEK INVALID TRACK
7644                          ;;*****************************************************************
7645  032476                  TST54:
7646  032476  000004                  SCOPE                   ;SCOPE CALL
7647  032500  000240                  NOP                     ;START OF TEST
7648  032502  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7649  032506  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7650  032512  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
 7651  032516  012737  000054  001226          MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 7652
 7653  032524  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7654  032530  054130                          .WORD   054130  ;TASK DESCRIPTOR
 7655  032532  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
 7656  032534  000240                          NOP                     ;RETURN HERE IF ERROR
 7657  032536  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7658  032540  000137  033112                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
 7659  032544  012737  000005  001376  80$:    MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
 7660  032552  012737  000000  001432          MOV     #0,RMDC0        ;DESIRED CYLINDER=0
 7661  032560  012737  002400  001404          MOV     #002400,RMDA0   ;SECTOR=0,TRACK=5
 7662  032566  012737  000000  001430          MOV     #0,RMOF0        ;18 BIT FORMAT
 7663  032574  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
 7664  032600  112722  000032                  MOVB    #RMOF,(R2)+     ;WRITE REGISTER INDEX
 7665  032604  112722  000034                  MOVB    #RMDC,(R2)+     ;TABLE FOR OUTPUT
 7666  032610  112722  000006                  MOVB    #RMDA,(R2)+
 7667  032614  112722  000000                  MOVB    #RMCS1,(R2)+
 7668  032620  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
 7669  032624  004737  043762                  JSR     PC,GETSTS       ;SETUP INPUT TABLE FOR STATUS
 7670  032630                          90$:
 7671  032630  004737  053210                  JSR     PC,CNTCLR
 7672  032634  000404                          BR      95$     ;GO TO 95$ IF NO ERROR
 7673  032636  000240                          NOP                     ;RETURN HERE IF ERROR
 7674  032640  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
 7675  032642  000137  033112                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7676  032646                          95$:
 7677  032646  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7678  032652  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
 7679  032654  000240                          NOP                     ;RETURN HERE IF ERROR
 7680  032656  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 7681  032660  000137  033112                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7682  032664                          100$:
 7683  032664  004737  053326                  JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
 7684  032670  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
 7685  032672  000240                          NOP                     ;RETURN HERE IF ERROR
 7686  032674  104000                          ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
 7687  032676  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7688  032700  000137  033112                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
 7689  032704                          110$:
 7690  032704  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7691  032710  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
 7692  032712  000240                          NOP                     ;RETURN HERE IF ERROR
 7693  032714  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 7694  032716  000137  033112                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7695  032722  004737  044656  120$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
 7696  032726  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7697  032732  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
 7698  032734  000240                          NOP                     ;RETURN HERE IF ERROR
 7699  032736  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 7700  032740  000137  033112                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7701  032744                          130$:
 7702  032744  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7703  032750  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
 7704  032752  000240                          NOP                     ;RETURN HERE IF ERROR
 7705  032754  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7706  032756  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
7707  032760  000137  033112                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
7708  032764                          140$:
7709  032764  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7710  032770  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7711  032772  000240                          NOP                     ;RETURN HERE IF ERROR
7712  032774  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7713  032776  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7714  033000  000137  033044                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7715  033004                          150$:
7716  033004                          160$:
7717  033004  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7718  033010  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
7719  033012  000010                          .WORD   DPE             ;MASK FOR RMER2
7720  033014  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
7721  033016  000240                          NOP                     ;RETURN HERE IF ERROR
7722  033020  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7723  033022  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7724  033024  000137  033044                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7725  033030                          170$:
7726  033030  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7727  033034  000403                          BR      180$                    ;GO TO 180$ IF NO ERROR
7728  033036  000240                          NOP                     ;RETURN HERE IF ERROR
7729  033040  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7730  033042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7731  033044  062737  000400  001404  180$:   ADD     #400,RMDAO      ;INCREMENT TRACK ADDRESS
7732  033052  022737  003400  001404          CMP     #003400,RMDAO   ;DONE??
7733  033060  103012                          BHIS    195$            ;NO-DO NEXT TRACK
7734  033062  032737  010000  001430  190$:   BIT     #FMT16,RMOFO    ;WAS TEST DONE FOR 16 BIT??
7735  033070  001010                          BNE     200$            ;YES!!
7736  033072  012737  010000  001430          MOV     #FMT16,RMOFO    ;SET FORMAT 16
7737  033100  012737  002400  001404          MOV     #002400,RMDAO   ;TRACK = 5
7738  033106  000137  032630          195$:   JMP     90$             ;DO TEST FOR 16 BIT FORMAT
7739  033112                          200$:
7740
7741                                  ;;**********************  ****************************************
7742                                  ;*TEST 55        SEEK INVALID CYLINDER
7743                                  ;;**************************************************************
7744  033112                          TST55:
7745  033112  000004                          SCOPE                   ;SCOPE CALL
7746  033114  000240                          NOP                     ;START OF TEST
7747  033116  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7748  033122  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7749  033126  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7750  033132  012737  000055  001226          MOV     #55,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7751
7752  033140  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7753  033144  054130                          .WORD   054130  ;TASK DESCRIPTOR
7754  033146  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
7755  033150  000240                          NOP                     ;RETURN HERE IF ERROR
7756  033152  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7757  033154  000137  033524                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
7758  033160  012737  000005  001376  80$:    MOV     #SEEK!GO,RMCS10 ;SEEK FUNCTION CODE
7759  033166  012737  000000  001404          MOV     #0,RMDAO        ;SECTOR = TRACK = 0
7760  033174  012737  001467  001432          MOV     #823.,RMDC0     ;CYLINDER = 823
7761  033202  012737  000000  001430          MOV     #0,RMOFO        ;18 BIT FORMAT
7762  033210  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
```

```
7763  033214  112722  000006                  MOVB    #RMDA,(R2)+     ;WRITE REGISTER TABLE
7764  033220  112722  000034                  MOVB    #RMDC,(R2)+
7765  033224  112722  000032                  MOVB    #RMOF,(R2)+
7766  033230  112722  000000                  MOVB    #RMCS1,(R2)+
7767  033234  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
7768  033240  004737  043762                  JSR     PC,GETSTS       ;SETUP FOR STATUS
7769  033244                          90$:
7770  033244  004737  053210                  JSR     PC,CNTCLR
7771  033250  000404                          BR      95$     ;GO TO 95$ IF NO ERROR
7772  033252  000240                          NOP                     ;RETURN HERE IF ERROR
7773  033254  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
7774  033256  000137  033460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7775  033262                          95$:
7776  033262  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7777  033266  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
7778  033270  000240                          NOP                     ;RETURN HERE IF ERROR
7779  033272  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7780  033274  000137  033460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7781  033300                          100$:
7782  033300  004737  053326                  JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
7783  033304  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7784  033306  000240                          NOP                     ;RETURN HERE IF ERROR
7785  033310  104000                          ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
7786  033312  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7787  033314  000137  033460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7788  033320                          110$:
7789  033320  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7790  033324  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
7791  033326  000240                          NOP                     ;RETURN HERE IF ERROR
7792  033330  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7793  033332  000137  033460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7794  033336  004737  044656          120$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
7795  033342  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7796  033346  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7797  033350  000240                          NOP                     ;RETURN HERE IF ERROR
7798  033352  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7799  033354  000137  033460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7800  033360                          130$:
7801  033360  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7802  033364  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
7803  033366  000240                          NOP                     ;RETURN HERE IF ERROR
7804  033370  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7805  033372  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7806  033374  000137  033460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7807  033400                          140$:
7808  033400  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7809  033404  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7810  033406  000240                          NOP                     ;RETURN HERE IF ERROR
7811  033410  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7812  033412  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7813  033414  000137  033460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7814  033420                          150$:
7815  033420                          160$:
7816  033420  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7817  033424  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
7818  033426  000010                          .WORD   DPE             ;MASK FOR RMER2
```

```
7819  033430  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
7820  033432  000240                        NOP                     ;RETURN HERE IF ERROR
7821  033434  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7822  033436  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7823  033440  000137  033460                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7824  033444                        170$:
7825  033444  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7826  033450  000403                        BR      180$                    ;GO TO 180$ IF NO ERROR
7827  033452  000240                        NOP                     ;RETURN HERE IF ERROR
7828  033454  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7829  033456  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7830  033460  005237  001432        180$:   INC     RMDC0           ;INCREMENT CYLINDER ADDRESS
7831  033464  022737  002000  001432        CMP     #1024.,RMDC0    ;DONE??
7832  033472  101012                        BHI     195$            ;NO-DO NEXT CYLINDER
7833  033474  032737  010000  001430  190$:   BIT     #FMT16,RMOF0    ;16 BIT FORMAT TESTED??
7834  033502  001010                        BNE     200$            ;YESS!!
7835  033504  012737  010000  001430        MOV     #FMT16,RMOF0    ;SETUP FOR 16 BIT TEST
7836  033512  012737  001467  001432        MOV     #823.,RMDC0
7837  033520  000137  033244        195$:   JMP     90$
7838  033524                        200$:
7839
7840                                ;;*****************************************************************
7841                                ;*TEST 56        IVC SEEK TEST
7842                                ;;*****************************************************************
7843  033524                        TST56:
7844  033524  000004                        SCOPE                   ;SCOPE CALL
7845  033526  000240                        NOP                     ;START OF TEST
7846  033530  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7847  033534  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7848  033540  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7849  033544  012737  000056  001226        MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7850
7851  033552  004737  053210                JSR     PC,CNTCLR
7852  033556  000404                        BR      10$     ;GO TO 10$ IF NO ERROR
7853  033560  000240                        NOP                     ;RETURN HERE IF ERROR
7854  033562  104000                        ERROR                   ;ERROR NUMBER DEFINED BY SUB
7855  033564  000137  034064                JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7856  033570                        10$:
7857  033570  112737  000024  001533        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7858  033576  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7859  033604  012737  000001  001422        MOV     #DMD,RMMR10     ;RMMR1=DMD
7860  033612  004737  044316                JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7861  033616  000404                        BR      20$             ;GO TO 20$ IF NO ERROR
7862  033620  000240                        NOP                     ;RETURN HERE IF ERROR
7863  033622  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7864  033624  000137  034064                JMP     90$             ;GO TO 90$ IF ERROR WAS FOUND
7865  033630                        20$:
7866  033630  012737  000005  001376        MOV     #SEEK!GO,RMCS10
7867  033636  012737  000000  001422        MOV     #0,RMMR10
7868  033644  012737  000000  001404        MOV     #0,RMDA0        ;SECTOR=TRACK=0
7869  033652  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER=0
7870  033660  012702  001533                MOV     #PUTINX,R2      ;WRITE REGISTER INDEX
7871  033664  112722  000024                MOVB    #RMMR1,(R2)+    ;TABLE
7872  033670  112722  000006                MOVB    #RMDA,(R2)+
7873  033674  112722  000034                MOVB    #RMDC,(R2)+
7874  033700  112722  000000                MOVB    #RMCS1,(R2)+
```

```
7875  033704  112722  000200                  MOVB    #200,(R2)+
7876  033710  004737  043762                  JSR     PC,GETSTS       ;SETUP FOR STATUS
7877  033714  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7878  033720  000404                          BR      30$     ;GO TO 30$ IF NO ERROR
7879  033722  000240                          NOP                     ;RETURN HERE IF ERROR
7880  033724  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7881  033726  000137  034064                  JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7882  033732                          30$:
7883  033732  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7884  033736  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
7885  033740  000240                          NOP                     ;RETURN HERE IF ERROR
7886  033742  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7887  033744  000137  034064                  JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7888  033750                          40$:
7889  033750  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7890  033754  000405                          BR      50$                     ;GO TO 50$ IF NO ERROR
7891  033756  000240                          NOP                     ;RETURN HERE IF ERROR
7892  033760  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7893  033762  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7894  033764  000137  034064                  JMP     90$                     ;GO TO 90$ IF ERROR WAS FOUND
7895  033770  032737  010000  001370  50$:    BIT     #IVC,RMER2I     ;DID INVALID COMMAND SET??
7896  033776  001012                          BNE     60$             ;YES!!
7897  034000  012737  010000  001140          MOV     #IVC,$GDDAT     ;GOOD DATA FOR TYPEOUT
7898  034006  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
7899  034014  042737  167777  001142          BIC     #^CIVC,$BDDAT
7900  034022  104064                          ERROR   64              ;IVC NOT DETECTED
7901  034024                          60$:
7902  034024                          70$:
7903  034024  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7904  034030  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
7905  034032  010010                          .WORD   IVC!DPE         ;MASK FOR RMER2
7906  034034  000405                          BR      80$                     ;GO TO 80$ IF NO ERROR
7907  034036  000240                          NOP                     ;RETURN HERE IF ERROR
7908  034040  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7909  034042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7910  034044  000137  034064                  JMP     90$                     ;GO TO 90$ IF ERROR WAS FOUND
7911  034050                          80$:
7912  034050  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7913  034054  000403                          BR      90$                     ;GO TO 90$ IF NO ERROR
7914  034056  000240                          NOP                     ;RETURN HERE IF ERROR
7915  034060  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7916  034062  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7917  034064                          90$:
7918
7919                                  ;;*****************************************************************
7920                                  ;*TEST 57        ABORT SEEK TEST
7921                                  ;;*****************************************************************
7922  034064                          TST57:
7923  034064  000004                          SCOPE                   ;SCOPE CALL
7924  034066  000240                          NOP                     ;START OF TEST
7925  034070  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7926  034074  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7927  034100  013701  001446                  MOV     TSTVUE,R1       ;(R1) = DEVICE BEING TESTED
7928  034104  012737  000057  001226          MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7929
7930  034112  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
 7931  034116  054130                          .WORD   054130  ;TASK DESCRIPTOR
 7932  034120  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
 7933  034122  000240                          NOP                     ;RETURN HERE IF ERROR
 7934  034124  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7935  034126  000137  034426                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 7936  034132  012737  040000  001412  80$:    MOV     #UNS,RMER10     ;SET UNSAFE ERROR
 7937  034140  012737  000000  001404          MOV     #0,RMDA0        ;SECTOR = TRACK = 0
```

```
7938  034146  012737  000000  001432          MOV     #0,RMDC0        ;CYLNDER = 0
7939  034154  012737  000005  001376          MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
7940  034162  012702  001533                  MOV     #PUTINX,R2      ;SETUP REGISTER INDEX TABLE
7941  034166  112722  000006                  MOVB    #RMDA,(R2)+     ;AND OUTPUT BUFFER
7942  034172  112722  000034                  MOVB    #RMDC,(R2)+
7943  034176  112722  000014                  MOVB    #RMER1,(R2)+
7944  034202  112722  000000                  MOVB    #RMCS1,(R2)+
7945  034206  112722  000200                  MOVB    #200,(R2)+
7946  034212  004737  043762                  JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7947  034216  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7948  034222  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
7949  034224  000240                          NOP                     ;RETURN HERE IF ERROR
7950  034226  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7951  034230  000137  034426                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7952  034234                          90$:
7953  034234  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7954  034240  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
7955  034242  000240                          NOP                     ;RETURN HERE IF ERROR
7956  034244  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7957  034246  000137  034426                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7958  034252  032737  020000  001340  100$:   BIT     #PIP,RMDSI      ;DID DRIVE START SEEK??
7959  034260  001411                          BEQ     110$            ;NO!!
7960  034262  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
7961  034270  042737  157777  001142          BIC     #^CPIP,$BDDAT
7962  034276  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
7963  034302  104065                          ERROR   65              ;DRIVE SHOULD NOT SEEK
7964  034304  004737  044656          110$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
7965  034310  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7966  034314  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
7967  034316  000240                          NOP                     ;RETURN HERE IF ERROR
7968  034320  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7969  034322  000137  034426                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7970  034326                          120$:
7971  034326  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7972  034332  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
7973  034334  000240                          NOP                     ;RETURN HERE IF ERROR
7974  034336  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7975  034340  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7976  034342  000137  034426                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7977  034346                          130$:
7978  034346  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
7979  034352  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
7980  034354  000240                          NOP                     ;RETURN HERE IF ERROR
7981  034356  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
7982  034360  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7983  034362  000137  034426                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7984  034366                          140$:
7985  034366  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7986  034372  155760                          .WORD   NOT^MSK!UNS             ;MASK FOR RMER1
7987  034374  000010                          .WORD   DPE             ;MASK FOR RMER2
7988  034376  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7989  034400  000240                          NOP                     ;RETURN HERE IF ERROR
7990  034402  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7991  034404  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7992  034406  000137  034426                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7993  034412                          150$:
```

```
                        T57     ABORT SEEK TEST

7994  034412  004737  045674            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7995  034416  000403                    BR      160$                    ;GO TO 160$ IF NO ERROR
7996  034420  000240                    NOP                     ;RETURN HERE IF ERROR
7997  034422  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7998  034424  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7999  034426                        160$:
8000
8001                                    ;;*****************************************************************
8002                                    ;*TEST 60       SEEK AT OFFSET
8003                                    ;;*****************************************************************
8004  034426                        TST60:
8005  034426  000004                    SCOPE                   ;SCOPE CALL
8006  034430  000240                    NOP                     ;START OF TEST
8007  034432  012737  000001  001206    MOV     #1,$TIMES       ;LOAD ITERATION COUNT
8008  034440  012737  034454  001122    MOV     #1$,$LPADR
8009  034446  012737  034454  001124    MOV     #1$,$LPERR
8010  034454                        1$:
8011  034454  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8012  034460  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8013  034464  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8014  034470  012737  000060  001226    MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8015  034476                        10$:
8016  034476  004737  043046            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8017  034502  054130                    .WORD   054130  ;TASK DESCRIPTOR
8018  034504  000404                    BR      20$                     ;GO TO 20$ IF NO ERROR
8019  034506  000240                    NOP                     ;RETURN HERE IF ERROR
8020  034510  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8021  034512  000137  034774            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
8022  034516                        20$:
8023  034516  012737  000000  001404    MOV     #0,RMDA0        ;TRACK = SECTOR = 0
8024  034524  012737  000000  001432    MOV     #0.,RMDC0       ;CYLINDER = 0
8025  034532  012737  000015  001376    MOV     #OFFSET!GO,RMCS10       ;LOAD OFFSET COMMAND IN BUFFER
8026  034540  012702  001533            MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
8027  034544  112722  000034            MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
8028  034550  112722  000006            MOVB    #RMDA,(R2)+
8029  034554  112722  000000            MOVB    #RMCS1,(R2)+
8030  034560  112722  000200            MOVB    #200,(R2)+      ;WRITE TERMINATOR
8031  034564  004737  044316            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8032  034570  000404                    BR      30$     ;GO TO 30$ IF NO ERROR
8033  034572  000240                    NOP                     ;RETURN HERE IF ERROR
8034  034574  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
8035  034576  000137  034774            JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8036  034602  004737  043762        30$:JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
8037  034606  004737  044656            JSR     PC,TIMOUT       ;WAIT FOR OFFSET TO COMPLETE
8038  034612                        40$:
8039  034612  012737  000005  001376    MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
8040  034620  112737  000000  001533    MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
8041  034626  112737  000200  001534    MOVB    #200,PUTINX+1
8042  034634  004737  044316            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8043  034640  000404                    BR      50$     ;GO TO 50$ IF NO ERROR
8044  034642  000240                    NOP                     ;RETURN HERE IF ERROR
8045  034644  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
8046  034646  000137  034774            JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8047  034652                        50$:
8048  034652  004737  044656            JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8049  034656                        60$:
```

```
 8050  034656  004737  044046          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8051  034662  000404                  BR      100$    ;GO TO 100$ IF NO ERROR
 8052  034664  000240                  NOP                     ;RETURN HERE IF ERROR
 8053  034666  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
 8054  034670  000137  034774          JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
 8055  034674                  100$:
 8056  034674  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8057  034700  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
 8058  034702  000240                  NOP                     ;RETURN HERE IF ERROR
 8059  034704  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8060  034706  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8061  034710  000137  034774          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
 8062  034714                  110$:
 8063  034714  004737  051750          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 8064  034720  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
 8065  034722  000240                  NOP                     ;RETURN HERE IF ERROR
 8066  034724  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 8067  034726  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8068  034730  000137  034774          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
 8069  034734                  120$:
 8070  034734  004737  061416          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 8071  034740  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
 8072  034742  000010                  .WORD   DPE             ;MASK FOR RMER2
 8073  034744  000405                  BR      130$                    ;GO TO 130$ IF NO ERROR
 8074  034746  000240                  NOP                     ;RETURN HERE IF ERROR
 8075  034750  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 8076  034752  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8077  034754  000137  034774          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
 8078  034760                  130$:
 8079  034760  004737  045674          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8080  034764  000403                  BR      140$                    ;GO TO 140$ IF NO ERROR
 8081  034766  000240                  NOP                     ;RETURN HERE IF ERROR
 8082  034770  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8083  034772  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8084  034774                  140$:
 8085                          ;;*****************************************************************
 8086                          ;*TEST 61        LOOK AHEAD TEST
 8087                          ;;*****************************************************************
 8088  034774                  TST61:
 8089  034774  000004                  SCOPE                   ;SCOPE CALL
 8090  034776  000240                  NOP                     ;START OF TEST
 8091  035000  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 8092  035004  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 8093  035010  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 8094  035014  012737  000061  001226  MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8095
 8096  035022  004737  053210          JSR     PC,CNTCLR
 8097  035026  000404                  BR      10$     ;GO TO 10$ IF NO ERROR
 8098  035030  000240                  NOP                     ;RETURN HERE IF ERROR
 8099  035032  104000                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
 8100  035034  000137  035446          JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8101  035040                  10$:
 8102  035040  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 8103  035044  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
 8104  035050  012737  035434  000004  MOV     #145$,ERRVEC    ;SETUP FOR BUS TIMOUT
 8105  035056  012737  000300  000006  MOV     #PR6,ERRVEC+2
```

```
 8106  035064  012703  000041                    MOV     #33.,R3          ;R3=SAMPLE COUNT
 8107  035070  012737  010000  001430            MOV     #FMT16,RMOF0     ;SETUP 16 BIT MODE
 8108  035076  012737  000000  001404            MOV     #0,RMDA0         ;SEARCH SECTOR 0
 8109  035104  012737  000000  001432            MOV     #0,RMDC0         ;CYLINDER 0
 8110  035112  012737  000031  001376            MOV     #SEARCH!GO,RMCS10        ;RECALIBRATE COMMAND
 8111  035120  012702  001533                    MOV     #PUTINX,R2
 8112  035124  112722  000006                    MOVB    #RMDA,(R2)+
 8113  035130  112722  000034                    MOVB    #RMDC,(R2)+
 8114  035134  112722  000032                    MOVB    #RMOF,(R2)+
 8115  035140  112722  000000                    MOVB    #RMCS1,(R2)+
 8116  035144  112722  000200                    MOVB    #200,(R2)+
 8117  035150  004737  043762                    JSR     PC,GETSTS
 8118  035154                            20$:
 8119  035154  004737  044316                    JSR     PC,PUT           ;GO WRITE REGISTERS VIA PUT SUB
 8120  035160  000404                            BR      30$      ;GO TO 30$ IF NO ERROR
 8121  035162  000240                            NOP                      ;RETURN HERE IF ERROR
 8122  035164  104000                            ERROR                    ;ERROR DEFINED BY PUT SUB
 8123  035166  000137  035436                    JMP     150$     ;GO TO 150$ IF ERROR WAS FOUND
 8124  035172  004737  044656            30$:    JSR     PC,TIMOUT        ;WAIT FOR COMPLETION
 8125  035176  004737  044046                    JSR     PC,GET           ;GO READ REGISTERS VIA GET SUB
 8126  035202  000404                            BR      40$      ;GO TO 40$ IF NO ERROR
 8127  035204  000240                            NOP                      ;RETURN HERE IF ERROR
 8128  035206  104000                            ERROR                    ;ERROR DEFINED BY GET SUB
 8129  035210  000137  035436                    JMP     150$     ;GO TO 150$ IF ERROR WAS FOUND
 8130  035214                            40$:
 8131  035214  004737  045042                    JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
 8132  035220  000405                            BR      50$              ;GO TO 50$ IF NO ERROR
 8133  035222  000240                            NOP                      ;RETURN HERE IF ERROR
 8134  035224  104000                            ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8135  035226  004736                            JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
 8136  035230  000137  035436                    JMP     150$             ;GO TO 150$ IF ERROR WAS FOUND
 8137  035234                            50$:
 8138  035234  012704  105642            60$:    MOV     #BUFFER,R4       ;R4=STORAGE ADDRESS
 8139  035240  012705  177777                    MOV     #-1,R5                   ;R5=GROSS TIMER
 8140  035244  016014  000020            70$:    MOV     RMLA(R0),(R4)    ;STORE RMLA SAMPLE
 8141  035250  016002  000020                    MOV     RMLA(R0),R2      ;GET ANOTHER LOOK
 8142  035254  020214                            CMP     R2,(R4)          ;ARE SAMPLES SAME??
 8143  035256  001403                            BEQ     80$              ;YES!!
 8144  035260  005305                            DEC     R5               ;TIMEOUT??
 8145  035262  001370                            BNE     70$              ;NO - TRY AGAIN
 8146  035264  000411                            BR      100$             ;PROGRAM TIMEOUT
 8147  035266  062704  000002            80$:    ADD     #2,R4            ;ADVANCE STORAGE ADDRESS
 8148  035272  005303                            DEC     R3               ;ALL SAMPLES TAKEN??
 8149  035274  001410                            BEQ     110$             ;YES!!
 8150  035276  026002  000020            90$:    CMP     RMLA(R0),R2      ;WAIT FOR CHANGE
 8151  035302  001360                            BNE     70$              ;YES!!
 8152  035304  005305                            DEC     R5               ;TIMEOUT??
 8153  035306  001373                            BNE     90$              ;NO
 8154  035310  104352                    100$:   ERROR   352              ;PROGRAM TIMEOUT
 8155  035312  000137  035446                    JMP     160$
 8156  035316  012703  000040            110$:   MOV     #32.,R3          ;R3 = NUMBER OF COMPARES
 8157  035322  012702  003700                    MOV     #3700,R2                 ;R2 = MAXIMUM SAMPLE
 8158  035326  012704  105642                    MOV     #BUFFER,R4       ;R4 = BUFFER ADDRESS
 8159  035332  032737  010000  001430            BIT     #FMT16,RMOF0     ;WAS SAMPLE FOR 16 BIT MODE??
 8160  035340  001004                            BNE     120$             ;YES
 8161  035342  012703  000036                    MOV     #30.,R3          ;CHANGE COMPARE COUNT AND
```

```
8162  035346  012702  003500                        MOV     #3500,R2                ;MAXIMUM SAMPLE
8163  035352  012405                        120$:   MOV     (R4)+,R5        ;GET A SAMPLE AND INCREMENT
8164  035354  062705  000100                        ADD     #100,R5         ;FOR EXPECTED VALUE OF NEXT
8165  035360  020205                                CMP     R2,R5           ;SHOULD NEXT BE 0??
8166  035362  103001                                BHIS    130$            ;NO!!
8167  035364  005005                                CLR     R5              ;YES - CHANGE EXPECTED
8168  035366  020514                        130$:   CMP     R5,(R4)         ;IS NEXT SAMPLE CORRECT??
8169  035370  001406                                BEQ     140$            ;YES!!
8170  035372  010537  001140                        MOV     R5,$GDDAT       ;EXPECTED VALUE
8171  035376  011437  001142                        MOV     (R4),$BDDAT     ;RECEIVED VALUE
8172  035402  104353                                ERROR   353             ;FAILURE IN LOOK AHEAD TEST
8173  035404  000414                                BR      150$
8174  035406  005303                        140$:   DEC     R3              ;ALL SAMPLES CHECKED??
8175  035410  001360                                BNE     120$            ;NO!!
8176  035412  032737  010000  001430                BIT     #FMT16,RMOFO    ;DONE TEST??
8177  035420  001406                                BEQ     150$            ;YES!!
8178  035422  005037  001430                        CLR     RMOFO           ;SETUP FOR 18 BIT MODE
8179  035426  012703  000037                        MOV     #31.,R3
8180  035432  000650                                BR      20$             ;DO AGAIN FOR 18 BIT MODE
8181  035434  022626                        145$:   CMP     (SP)+,(SP)+     ;RESTORE STACK
8182  035436                                150$:
8183  035436  012637  000006                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8184  035442  012637  000004                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8185  035446                                160$:
8186                                        ;;*****************************************************************
8187                                        ;*TEST 62       SEARCH ON CYLINDER
8188                                        ;;*****************************************************************
8189  035446                                TST62:
8190  035446  000004                                SCOPE                           ;SCOPE CALL
8191  035450  000240                                NOP                             ;START OF TEST
8192  035452  012706  001100                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
8193  035456  013700  001276                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
8194  035462  013701  001446                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
8195  035466  012737  000062  001226                MOV     #62,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
8196  035474  004737  043046                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8197  035500  054130                                .WORD   054130  ;TASK DESCRIPTOR
8198  035502  000404                                BR      80$                             ;GO TO 80$ IF NO ERROR
8199  035504  000240                                NOP                             ;RETURN HERE IF ERROR
8200  035506  104000                                ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8201  035510  000137  036124                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
8202  035514  012703  000035                80$:    MOV     #29.,R3                 ;R3=MAXIMUM SECTOR ADDRESS
8203  035520  012737  000000  001404                MOV     #0,RMDAO                ;TRACK=SECTOR=0
8204  035526  012737  000000  001432                MOV     #0,RMDCO                ;CYLINDER=0
8205  035534  012737  000000  001430                MOV     #0,RMOFO                ;18 BIT FORMAT
8206  035542  012737  000005  001376  85$:    MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
8207  035550  012702  001533                        MOV     #PUTINX,R2              ;SETUP REGISTER INDEX TABLE
8208  035554  112722  000006                        MOVB    #RMDA,(R2)+             ;FOR SEEK TO CYLINDER 0
8209  035560  112722  000034                        MOVB    #RMDC,(R2)+
8210  035564  112722  000032                        MOVB    #RMOF,(R2)+
8211  035570  112722  000000                        MOVB    #RMCS1,(R2)+
8212  035574  112722  000200                        MOVB    #200,(R2)+
8213  035600  004737  043762                        JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
8214  035604  004737  044316                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
8215  035610  000404                                BR      90$     ;GO TO 90$ IF NO ERROR
8216  035612  000240                                NOP                             ;RETURN HERE IF ERROR
8217  035614  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
```

```
8218  035616  000137  036124            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8219  035622  004737  044656    90$:    JSR     PC,TIMOUT
8220  035626  004737  044046            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8221  035632  000404                    BR      100$    ;GO TO 100$ IF NO ERROR
8222  035634  000240                    NOP                     ;RETURN HERE IF ERROR
8223  035636  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
8224  035640  000137  036124            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8225  035644                    100$:
8226  035644  004737  045042            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8227  035650  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
8228  035652  000240                    NOP                     ;RETURN HERE IF ERROR
8229  035654  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8230  035656  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8231  035660  000137  036124            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8232  035664                    110$:
8233  035664  004737  051750            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
8234  035670  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
8235  035672  000240                    NOP                     ;RETURN HERE IF ERROR
8236  035674  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
8237  035676  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8238  035700  000137  036124            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8239  035704  012737  000031  001376  120$:    MOV     #SEARCH!GO,RMCS1;STORE SEARCH COMMAND
8240  035712  004737  044316            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8241  035716  000404                    BR      130$    ;GO TO 130$ IF NO ERROR
8242  035720  000240                    NOP                     ;RETURN HERE IF ERROR
8243  035722  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
8244  035724  000137  036124            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8245  035730                    130$:
8246  035730  004737  044656            JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
8247  035734  004737  044046            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8248  035740  000404                    BR      140$    ;GO TO 140$ IF NO ERROR
8249  035742  000240                    NOP                     ;RETURN HERE IF ERROR
8250  035744  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
8251  035746  000137  036124            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8252  035752                    140$:
8253  035752  004737  045042            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8254  035756  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
8255  035760  000240                    NOP                     ;RETURN HERE IF ERROR
8256  035762  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8257  035764  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8258  035766  000137  036124            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8259  035772                    150$:
8260  035772  004737  057346            JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
8261  035776  000405                    BR      160$                    ;GO TO 160$ IF NO ERROR
8262  036000  000240                    NOP                     ;RETURN HERE IF ERROR
8263  036002  104000                    ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
8264  036004  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8265  036006  000137  036124            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8266  036012                    160$:
8267  036012  004737  061416            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8268  036016  115760                    .WORD   NDTMSK          ;MASK FOR RMER1
8269  036020  000010                    .WORD   DPE             ;MASK FOR RMER2
8270  036022  000405                    BR      170$                    ;GO TO 170$ IF NO ERROR
8271  036024  000240                    NOP                     ;RETURN HERE IF ERROR
8272  036026  104000                    ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8273  036030  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
 8274  036032  000137  036124                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8275  036036                            170$:
 8276  036036  004737  045674                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8277  036042  000405                            BR      180$                    ;GO TO 180$ IF NO ERROR
 8278  036044  000240                            NOP                     ;RETURN HERE IF ERROR
 8279  036046  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8280  036050  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8281  036052  000137  036124                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8282  036056  005237  001404            180$:   INC     RMDA0           ;INCREMENT SECTOR ADDRESS
 8283  036062  020337  001404                    CMP     R3,RMDA0        ;DONE ALL SECTORS??
 8284  036066  103306                            BHIS    120$            ;NO!!
 8285  036070  032737  010000  001430            BIT     #FMT16,RMOF0    ;IS 16 BIT FORMAT DONE??
 8286  036076  001012                            BNE     190$            ;YES!!
 8287  036100  012737  010000  001430            MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
 8288  036106  012703  000037                    MOV     #31.,R3         ;R3=MAXIMUM SECTOR ADDRESS
 8289  036112  012737  000000  001404            MOV     #0,RMDA0        ;START AT SECTOR 0
 8290  036120  000137  035542                    JMP     85$             ;GO TEST 16 BIT MODE
 8291  036124                            190$:
 8292
 8293
 8294
 8295                                      ;;*****************************************************************
 8296                                      ;*TEST 63       SEARCH OFF CYLINDER
 8297                                      ;;*****************************************************************
 8298  036124                            TST63:
 8299  036124  000004                            SCOPE                   ;SCOPE CALL
 8300  036126  000240                            NOP                     ;START OF TEST
 8301  036130  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 8302  036134  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 8303  036140  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 8304  036144  012737  000063  001226            MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8305  036152  004737  043046                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 8306  036156  054130                            .WORD   054130  ;TASK DESCRIPTOR
 8307  036160  000404                            BR      80$                     ;GO TO 80$ IF NO ERROR
 8308  036162  000240                            NOP                     ;RETURN HERE IF ERROR
 8309  036164  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 8310  036166  000137  036632                    JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
 8311  036172  012737  000000  001404  80$:      MOV     #0,RMDA0
 8312  036200  012737  000000  001430            MOV     #0,RMOF0        ;START WITH 18 BIT MODE
 8313  036206  012704  000035                    MOV     #29.,R4         ;R4=MAXIMUM SECTOR ADDRESS
 8314  036212  012703  000633                    MOV     #411.,R3        ;R3=SEARCH CYLINDER
 8315  036216  012705  000634                    MOV     #412.,R5        ;R5=SEEK CYLINDER
 8316  036222  012737  000005  001376  90$:      MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
 8317  036230  010537  001432                    MOV     R5,RMDC0        ;LOAD CYLINDER ADDRESS
 8318  036234  012702  001533                    MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
 8319  036240  112722  000032                    MOVB    #RMOF,(R2)+     ;SETUP REGISTER INDEX TABLE
 8320  036244  112722  000034                    MOVB    #RMDC,(R2)+     ;FOR SEEK COMMAND
 8321  036250  112722  000006                    MOVB    #RMDA,(R2)+
 8322  036254  112722  000000                    MOVB    #RMCS1,(R2)+
 8323  036260  112722  000200                    MOVB    #200,(R2)+      ;TERMINATE TABLE
 8324  036264  004737  043762                    JSR     PC,GETSTS       ;SETUP FOR STATUS
 8325  036270  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 8326  036274  000404                            BR      100$    ;GO TO 100$ IF NO ERROR
 8327  036276  000240                            NOP                     ;RETURN HERE IF ERROR
 8328  036300  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
 8329  036302  000137  036632                    JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
```

```
 8330  036306  004737  044656          100$:  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
 8331  036312  004737  044046                 JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8332  036316  000404                         BR      110$    ;GO TO 110$ IF NO ERROR
 8333  036320  000240                         NOP                     ;RETURN HERE IF ERROR
 8334  036322  104000                         ERROR                   ;ERROR DEFINED BY GET SUB
 8335  036324  000137  036632                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
 8336  036330                         110$:
 8337  036330  004737  045042                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8338  036334  000405                         BR      120$                    ;GO TO 120$ IF NO ERROR
 8339  036336  000240                         NOP                     ;RETURN HERE IF ERROR
 8340  036340  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8341  036342  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8342  036344  000137  036632                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
 8343  036350                         120$:
 8344  036350  004737  051750                 JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 8345  036354  000405                         BR      130$                    ;GO TO 130$ IF NO ERROR
 8346  036356  000240                         NOP                     ;RETURN HERE IF ERROR
 8347  036360  104000                         ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 8348  036362  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8349  036364  000137  036632                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
 8350  036370  010337  001432          130$:  MOV     R3,RMDC0        ;LOAD CYLINDER ADDRESS
 8351  036374  012737  000031  001376         MOV     #SEARCH!GO,RMCS1;LOAD SEARCH COMMAND
 8352  036402  004737  044316                 JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 8353  036406  000404                         BR      140$    ;GO TO 140$ IF NO ERROR
 8354  036410  000240                         NOP                     ;RETURN HERE IF ERROR
 8355  036412  104000                         ERROR                   ;ERROR DEFINED BY PUT SUB
 8356  036414  000137  036632                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
 8357  036420  004737  044656          140$:  JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
 8358  036424  004737  044046                 JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8359  036430  000404                         BR      150$    ;GO TO 150$ IF NO ERROR
 8360  036432  000240                         NOP                     ;RETURN HERE IF ERROR
 8361  036434  104000                         ERROR                   ;ERROR DEFINED BY GET SUB
 8362  036436  000137  036632                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
 8363  036442                         150$:
 8364  036442  004737  045042                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8365  036446  000405                         BR      160$                    ;GO TO 160$ IF NO ERROR
 8366  036450  000240                         NOP                     ;RETURN HERE IF ERROR
 8367  036452  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8368  036454  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8369  036456  000137  036632                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
 8370  036462                         160$:
 8371  036462  004737  057346                 JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
 8372  036466  000405                         BR      170$                    ;GO TO 170$ IF NO ERROR
 8373  036470  000240                         NOP                     ;RETURN HERE IF ERROR
 8374  036472  104000                         ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
 8375  036474  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8376  036476  000137  036632                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
 8377  036502                         170$:
 8378  036502  004737  061416                 JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 8379  036506  115760                         .WORD   NDTMSK          ;MASK FOR RMER1
 8380  036510  000010                         .WORD   DPE             ;MASK FOR RMER2
 8381  036512  000405                         BR      180$                    ;GO TO 180$ IF NO ERROR
 8382  036514  000240                         NOP                     ;RETURN HERE IF ERROR
 8383  036516  104000                         ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 8384  036520  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8385  036522  000137  036632                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
```

```
 8386  036526                                  180$:
 8387  036526  004737  045674                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8388  036532  000405                                BR      190$                    ;GO TO 190$ IF NO ERROR
 8389  036534  000240                                NOP                     ;RETURN HERE IF ERROR
 8390  036536  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8391  036540  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8392  036542  000137  036546                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8393  036546  005303                          190$: DEC     R3              ;DECREMENT SEARCH CYLINDER
 8394  036550  005205                                INC     R5              ;INCREMENT SEEK CYLINDER
 8395  036552  005237  001404                        INC     RMDA0           ;INCREMENT SECTOR ADDRESS
 8396  036556  020437  001404                        CMP     R4,RMDA0        ;DONE ALL SECTORS??
 8397  036562  103401                                BLO     210$            ;YES!!
 8398  036564  000616                                BR      90$             ;DO NEXT SECTOR
 8399  036566  032737  010000  001430          210$: BIT     #FMT16,RMOF0    ;DONE BOTH FORMATS??
 8400  036574  001016                                BNE     230$            ;YES!!
 8401  036576  012704  000037                        MOV     #31.,R4         ;R4=MAXIMUM SECTOR ADDRESS
 8402  036602  012703  000634                        MOV     #412.,R3        ;R3=SEARCH CYLINDER
 8403  036606  012705  000633                        MOV     #411.,R5        ;R5=SEEK CYLINDER
 8404  036612  012737  010000  001430                MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
 8405  036620  012737  000000  001404                MOV     #0,RMDA0        ;START AT SECTOR 0
 8406  036626  000137  036222                        JMP     90$             ;START 16 BIT TEST
 8407  036632                                  230$:
 8408                                          ;;*****************************************************************
 8409                                          ;*TEST 64       SEARCH INVALID SECTOR
 8410                                          ;;*****************************************************************
 8411  036632                                  T$T64:
 8412  036632  000004                                SCOPE                   ;SCOPE CALL
 8413  036634  000240                                NOP                     ;START OF TEST
 8414  036636  012706  001100                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 8415  036642  013700  001276                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 8416  036646  013701  001446                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 8417  036652  012737  000064  001226                MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8418
 8419  036660  004737  043046                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 8420  036664  054130                                .WORD   054130  ;TASK DESCRIPTOR
 8421  036666  000404                                BR      80$                     ;GO TO 80$ IF NO ERROR
 8422  036670  000240                                NOP                     ;RETURN HERE IF ERROR
 8423  036672  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 8424  036674  000137  037212                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8425  036700  012737  000000  001430  80$:   MOV     #0,RMOF0        ;SET 18 BIT FORMAT
 8426  036706  012737  000036  001404                MOV     #30.,RMDA0      ;START WITH SECTOR 30
 8427  036714  012737  000000  001432                MOV     #0,RMDC0        ;CYLINDER=0
 8428  036722  012737  000031  001376                MOV     #SEARCH!GO,RMCS1;SEARCH COMMAND
 8429  036730  012702  001533                        MOV     #PUTINX,R2      ;WRITE REGISTER TABLE FOR
 8430  036734  112722  000034                        MOVB    #RMDC,(R2)+     ;COMMAND
 8431  036740  112722  000032                        MOVB    #RMOF,(R2)+
 8432  036744  112722  000006                        MOVB    #RMDA,(R2)+
 8433  036750  112722  000000                        MOVB    #RMCS1,(R2)+
 8434  036754  112722  000200                        MOVB    #200,(R2)+      ;TERMINATE TABLE
 8435  036760  004737  043762                        JSR     PC,GETSTS       ;SETUP STATUS FETCH
 8436  036764                                  90$:
 8437  036764  004737  044316                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 8438  036770  000404                                BR      100$    ;GO TO 100$ IF NO ERROR
 8439  036772  000240                                NOP                     ;RETURN HERE IF ERROR
 8440  036774  104000                                ERROR                   ;ERROR DEFINED BY PUT SUB
 8441  036776  000137  037160                        JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
```

```
8442  037002  004737  044656          100$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8443  037006  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8444  037012  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
8445  037014  000240                          NOP                     ;RETURN HERE IF ERROR
8446  037016  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
8447  037020  000137  037160                  JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
8448  037024                          120$:
8449  037024  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8450  037030  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
8451  037032  000240                          NOP                     ;RETURN HERE IF ERROR
8452  037034  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8453  037036  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8454  037040  000137  037160                  JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8455  037044  032737  002000  001342  130$:   BIT     #IAE,RMER1I     ;DID "IAE" SET??
8456  037052  001012                          BNE     140$            ;YES!!
8457  037054  012737  002000  001140          MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
8458  037062  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8459  037070  042737  175777  001142          BIC     #^CIAE,$BDDAT
8460  037076  104257                          ERROR   257             ;IVC NOT DETECTED
8461  037100                          140$:
8462  037100  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
8463  037104  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8464  037106  000240                          NOP                     ;RETURN HERE IF ERROR
8465  037110  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
8466  037112  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8467  037114  000137  037160                  JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8468  037120                          150$:
8469  037120  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8470  037124  117760                          .WORD   NOT^MSK!IAE             ;MASK FOR RMER1
8471  037126  000010                          .WORD   DPE             ;MASK FOR RMER2
8472  037130  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
8473  037132  000240                          NOP                     ;RETURN HERE IF ERROR
8474  037134  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8475  037136  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8476  037140  000137  037160                  JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8477  037144                          160$:
8478  037144  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8479  037150  000403                          BR      170$                    ;GO TO 170$ IF NO ERROR
8480  037152  000240                          NOP                     ;RETURN HERE IF ERROR
8481  037154  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8482  037156  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8483  037160  005237  001404          170$:   INC     RMDAO           ;INCREMENT SECTOR ADDRESS
8484  037164  022737  000037  001404          CMP     #31.,RMDAO      ;DONE??
8485  037172  103407                          BLO     190$            ;YES!!
8486  037174  004737  053210                  JSR     PC,CNTCLR
8487  037200  000402                          BR      180$    ;GO TO 180$ IF NO ERROR
8488  037202  000240                          NOP                     ;RETURN HERE IF ERROR
8489  037204  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
8490  037206  000137  036764          180$:   JMP     90$             ;TEST NEXT SECTOR
8491  037212                          190$:
8492
8493
8494
8495
8496
8497
```

```
8498
8499
8500
8501
8502
8503
8504                                    ;;*****************************************************************
8505                                    ;*TEST 65       SEARCH INVALID TRACK
8506                                    ;;*****************************************************************
8507  037212                            TST65:
8508  037212  000004                            SCOPE                   ;SCOPE CALL
8509  037214  000240                            NOP                     ;START OF TEST
8510  037216  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8511  037222  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8512  037226  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8513  037232  012737  000065  001226            MOV     #65,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
8514  037240  004737  043046                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8515  037244  054130                            .WORD   054130  ;TASK DESCRIPTOR
8516  037246  000404                            BR      80$             ;GO TO 80$ IF NO ERROR
8517  037250  000240                            NOP                     ;RETURN HERE IF ERROR
8518  037252  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8519  037254  000137  037600                    JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
8520
8521  037260  012737  000000  001430  80$:     MOV     #0,RMOF0        ;SET 18 BIT FORMAT
8522  037266  012737  002400  001404            MOV     #002400,RMDA0   ;START WITH TRACK 5, SECTOR 0
8523  037274  012737  000000  001432            MOV     #0,RMDC0        ;CYLINDER=0
8524  037302  012737  000031  001376            MOV     #SEARCH!GO,RMCS10;SEARCH COMMAND
8525  037310  012702  001533                    MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
8526  037314  112722  000006                    MOVB    #RMDA,(R2)+
8527  037320  112722  000034                    MOVB    #RMDC,(R2)+
8528  037324  112722  000032                    MOVB    #RMOF,(R2)+
8529  037330  112722  000000                    MOVB    #RMCS1,(R2)+
8530  037334  112722  000200                    MOVB    #200,(R2)+      ;TERMINATE TABLE
8531  037340  004737  043762                    JSR     PC,GETSTS
8532  037344                            90$:
8533  037344  004737  044316                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8534  037350  000404                            BR      100$    ;GO TO 100$ IF NO ERROR
8535  037352  000240                            NOP                     ;RETURN HERE IF ERROR
8536  037354  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
8537  037356  000137  037520                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8538  037362  004737  044656            100$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8539  037366  004737  044046                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8540  037372  000404                            BR      110$    ;GO TO 110$ IF NO ERROR
8541  037374  000240                            NOP                     ;RETURN HERE IF ERROR
8542  037376  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
8543  037400  000137  037520                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8544  037404                            110$:
8545  037404  004737  045042                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8546  037410  000405                            BR      120$            ;GO TO 120$ IF NO ERROR
8547  037412  000240                            NOP                     ;RETURN HERE IF ERROR
8548  037414  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8549  037416  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8550  037420  000137  037520                    JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8551  037424  032737  002000  001342  120$:     BIT     #IAE,RMER1I     ;DID "IAE" SET??
8552  037432  001012                            BNE     130$            ;YES!!
8553  037434  012737  002000  001140            MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
```

```
8554  037442  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8555  037450  042737  175777  001142          BIC     #^CIAE,$BDDAT
8556  037456  104257                          ERROR   257             ;IAE NOT SET DURING SEARCH
8557  037460                          130$:
8558  037460                          140$:
8559  037460  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8560  037464  117760                          .WORD   NOT^MSK!IAE             ;MASK FOR RMER1
8561  037466  000010                          .WORD   DPE             ;MASK FOR RMER2
8562  037470  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8563  037472  000240                          NOP                     ;RETURN HERE IF ERROR
8564  037474  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8565  037476  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8566  037500  000137  037520                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8567  037504                          150$:
8568  037504  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8569  037510  000403                          BR      160$                    ;GO TO 160$ IF NO ERROR
8570  037512  000240                          NOP                     ;RETURN HERE IF ERROR
8571  037514  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8572  037516  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8573  037520  062737  000400  001404  160$:   ADD     #400,RMDAO      ;INCREMENT TRACK ADDRESS
8574  037526  022737  004000  001404          CMP     #4000,RMDAO     ;DONE ALL TRACKS??
8575  037534  101012                          BHI     170$            ;NO!!
8576  037536  032737  010000  001430          BIT     #FMT16,RMOF0    ;DONE 16 BIT MODE??
8577  037544  001015                          BNE     180$            ;YES!!
8578  037546  012737  010000  001430          MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
8579  037554  012737  002400  001404          MOV     #002400,RMDAO   ;START WITH TRACK 5
8580  037562                          170$:                           ;TEST NEXT TRACK
8581  037562  004737  053210                  JSR     PC,CNTCLR
8582  037566  000402                          BR      175$    ;GO TO 175$ IF NO ERROR
8583  037570  000240                          NOP                     ;RETURN HERE IF ERROR
8584  037572  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
8585  037574  000137  037344          175$:   JMP     90$
8586
8587  037600                          180$:
8588
8589
8590
8591                                    ;;*****************************************************************
8592                                    ;*TEST 66        SEARCH INVALID CYLINDER
8593                                    ;;*****************************************************************
8594  037600                          TST66:
8595  037600  000004                          SCOPE                   ;SCOPE CALL
8596  037602  000240                          NOP                     ;START OF TEST
8597  037604  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8598  037610  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8599  037614  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8600  037620  012737  000066  001226          MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8601
8602  037626  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8603  037632  054130                          .WORD   054130  ;TASK DESCRIPTOR
8604  037634  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8605  037636  000240                          NOP                     ;RETURN HERE IF ERROR
8606  037640  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8607  037642  000137  040204                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8608  037646  012737  000000  001430  80$:    MOV     #0,RMOF0        ;SET 18 BIT FORMAT
8609  037654  012737  000000  001404          MOV     #0,RMDAO        ;TRACK=SECTOR=0
```

```
8610  037662  012737  001467  001432          MOV     #823.,RMDCO     ;START AT CYLINDER  823
8611  037670  012737  000031  001376          MOV     #SEARCH!GO,RMCS1;SEARCH COMMAND
8612  037676  012702  001533                  MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
8613  037702  112722  000032                  MOVB    #RMOF,(R2)+
8614  037706  112722  000034                  MOVB    #RMDC,(R2)+
8615  037712  112722  000006                  MOVB    #RMDA,(R2)+
8616  037716  112722  000000                  MOVB    #RMCS1,(R2)+
8617  037722  112722  000200                  MOVB    #200,(R2)+
8618  037726  004737  043762                  JSR     PC,GETSTS       ;SETUP STATUS FETCH
8619  037732                          90$:
8620  037732  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8621  037736  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
8622  037740  000240                          NOP                     ;RETURN HERE IF ERROR
8623  037742  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
8624  037744  000137  040126                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8625  037750  004737  044656          100$:   JSR     PC,TIMOUT
8626  037754  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8627  037760  000404                          BR      110$    ;GO TO 110$ IF NO ERROR
8628  037762  000240                          NOP                     ;RETURN HERE IF ERROR
8629  037764  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
8630  037766  000137  040126                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8631  037772                          110$:
8632  037772  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8633  037776  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
8634  040000  000240                          NOP                     ;RETURN HERE IF ERROR
8635  040002  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8636  040004  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8637  040006  000137  040126                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8638  040012  032737  002000  001342  120$:   BIT     #IAE,RMER1I     ;IS "IAE" SET??
8639  040020  001012                          BNE     130$            ;YES!!
8640  040022  012737  002000  001140          MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
8641  040030  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8642  040036  042737  175777  001142          BIC     #^CIAE,$BDDAT
8643  040044  104257                          ERROR   257             ;IAE NOT SET DURING SEARCH
8644  040046                          130$:
8645  040046  004737  060712                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
8646  040052  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
8647  040054  000240                          NOP                     ;RETURN HERE IF ERROR
8648  040056  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
8649  040060  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8650  040062  000137  040126                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8651  040066                          140$:
8652  040066  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8653  040072  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
8654  040074  000010                          .WORD   DPE             ;MASK FOR RMER2
8655  040076  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8656  040100  000240                          NOP                     ;RETURN HERE IF ERROR
8657  040102  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8658  040104  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8659  040106  000137  040126                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8660  040112                          150$:
8661  040112  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8662  040116  000403                          BR      160$                    ;GO TO 160$ IF NO ERROR
8663  040120  000240                          NOP                     ;RETURN HERE IF ERROR
8664  040122  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8665  040124  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
8666  040126  005237  001432          160$:   INC     RMDC0           ;INCREMENT CYLINDER ADDRESS
8667  040132  022737  002000  001432          CMP     #1024.,RMDC0    ;DONE ALL CYLINDERS??
8668  040140  101012                          BHI     170$            ;NO!!
8669  040142  032737  010000  001430          BIT     #FMT16,RMOF0    ;SET 16 BIT FORMAT
8670  040150  001015                          BNE     190$            ;YES!!
8671  040152  012737  010000  001430          MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
8672  040160  012737  001467  001432          MOV     #823.,RMDC0     ;CYLINDER=823
8673  040166                          170$:
8674  040166  004737  053210                  JSR     PC,CNTCLR
8675  040172  000402                          BR      180$    ;GO TO 180$ IF NO ERROR
8676  040174  000240                          NOP                     ;RETURN HERE IF ERROR
8677  040176  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
8678  040200  000137  037732          180$:   JMP     90$
8679  040204                          190$:
8680
8681
8682
8683
8684
8685                                    ;;*****************************************************************
8686                                    ;*TEST 67       IVC SEARCH TEST
8687                                    ;;*****************************************************************
8688  040204                          TST67:
8689  040204  000004                          SCOPE                   ;SCOPE CALL
8690  040206  000240                          NOP                     ;START OF TEST
8691  040210  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8692  040214  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8693  040220  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8694  040224  012737  000067  001226          MOV     #67,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8695
8696  040232  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8697  040236  054130                          .WORD   054130  ;TASK DESCRIPTOR
8698  040240  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8699  040242  000240                          NOP                     ;RETURN HERE IF ERROR
8700  040244  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8701  040246  000137  040564                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8702  040252                          80$:
8703  040252  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
8704  040260  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
8705  040266  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
8706  040274  004737  044316                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
8707  040300  000404                          BR      90$             ;GO TO 90$ IF NO ERROR
8708  040302  000240                          NOP                     ;RETURN HERE IF ERROR
8709  040304  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
8710  040306  000137  040564                  JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8711  040312  012737  000000  001422  90$:     MOV     #0,RMMR10       ;RESET DIAGNOSTIC MODE
8712  040320  012737  000000  001432          MOV     #0,RMDC0        ;CYLINDER 0
8713  040326  012737  000000  001404          MOV     #0,RMDA0        ;TRACK=SECTOR=0
8714  040334  012737  000000  001430          MOV     #0,RMOF0        ;18 BIT FORMAT
8715  040342  012737  000031  001376          MOV     #SEARCH!GO,RMCS10;SEARCH COMMAND
8716  040350  012702  001533                  MOV     #PUTINX,R2      ;LOAD INDEX TABLE
8717  040354  112722  000024                  MOVB    #RMMR1,(R2)+
8718  040360  112722  000034                  MOVB    #RMDC,(R2)+
8719  040364  112722  000006                  MOVB    #RMDA,(R2)+
8720  040370  112722  000032                  MOVB    #RMOF,(R2)+
8721  040374  112722  000000                  MOVB    #RMCS1,(R2)+
```

```
8722  040400  112722  000200                  MOVB    #200,(R2)+
8723  040404  004737  043762                  JSR     PC,GETSTS
8724  040410  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8725  040414  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
8726  040416  000240                          NOP                     ;RETURN HERE IF ERROR
8727  040420  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
8728  040422  000137  040564                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8729  040426  004737  044656          100$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8730  040432  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8731  040436  000404                          BR      110$    ;GO TO 110$ IF NO ERROR
8732  040440  000240                          NOP                     ;RETURN HERE IF ERROR
8733  040442  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
8734  040444  000137  040564                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8735  040450                          110$:
8736  040450  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8737  040454  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
8738  040456  000240                          NOP                     ;RETURN HERE IF ERROR
8739  040460  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8740  040462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8741  040464  000137  040564                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8742  040470  032737  010000  001370  120$:   BIT     #IVC,RMER2I     ;DID IVC SET??
8743  040476  001012                          BNE     130$            ;YES!!
8744  040500  012737  010000  001140          MOV     #IVC,$GDDAT     ;EXPECTED STATUS
8745  040506  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
8746  040514  042737  167777  001142          BIC     #^CIVC,$BDDAT
8747  040522  104260                          ERROR   260             ;IVC NOT SET DURING SEARCH
8748  040524                          130$:
8749  040524                          140$:
8750  040524  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8751  040530  115760                          .WORD   NOTMSK          ;MASK FOR RMER1
8752  040532  010010                          .WORD   IVC!DPE         ;MASK FOR RMER2
8753  040534  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8754  040536  000240                          NOP                     ;RETURN HERE IF ERROR
8755  040540  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8756  040542  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8757  040544  000137  040564                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8758  040550                          150$:
8759  040550  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8760  040554  000403                          BR      160$                    ;GO TO 160$ IF NO ERROR
8761  040556  000240                          NOP                     ;RETURN HERE IF ERROR
8762  040560  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8763  040562  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9764  040564                          160$:
8765                                  ;;*****************************************************************
8766                                  ;*TEST 70        ABORT SEARCH TEST
8767                                  ;;*****************************************************************
8768  040564                          TST70:
8769  040564  000004                          SCOPE                   ;SCOPE CALL
8770  040566  000240                          NOP                     ;START OF TEST
8771  040570  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8772  040574  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8773  040600  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8774  040604  012737  000070  001226          MOV     #70,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8775
8776  040612  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8777  040616  054130                          .WORD   054130  ;TASK DESCRIPTOR
```

```
 8778  040620  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
 8779  040622  000240                        NOP                             ;RETURN HERE IF ERROR
 8780  040624  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 8781  040626  000137  041130                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8782  040632  012737  040000  001412  80$:  MOV     #UNS,RMER10             ;SET UNSAFE ERROR
 8783  040640  012737  000031  001376        MOV     #SEARCH!GO,RMCS10       ;SEARCH COMMAND
 8784  040646  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER 0
 8785  040654  012737  000000  001404        MOV     #0,RMDA0                ;SECTOR=TRACK=0
 8786  040662  012702  001533                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 8787  040666  112722  000034                MOVB    #RMDC,(R2)+
 8788  040672  112722  000006                MOVB    #RMDA,(R2)+
 8789  040676  112722  000014                MOVB    #RMER1,(R2)+
 8790  040702  112722  000000                MOVB    #RMCS1,(R2)+
 8791  040706  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
 8792  040712  004737  043762                JSR     PC,GETSTS               ;SETUP FOR STATUS
 8793  040716  004737  044316                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 8794  040722  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
 8795  040724  000240                        NOP                             ;RETURN HERE IF ERROR
 8796  040726  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
 8797  040730  000137  041130                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8798  040734                        90$:
 8799  040734  004737  044046                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 8800  040740  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
 8801  040742  000240                        NOP                             ;RETURN HERE IF ERROR
 8802  040744  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
 8803  040746  000137  041130                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8804  040752  032737  020000  001340  100$:  BIT     #PIP,RMDSI              ;IS PIP SET??
 8805  040760  001412                        BEQ     110$                    ;NO!!
 8806  040762  012737  000000  001140        MOV     #0,$GDDAT               ;EXPECTED STATUS
 8807  040770  013737  001340  001142        MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
 8808  040776  042737  157777  001142        BIC     #^CPIP,$BDDAT
 8809  041004  104261                        ERROR   261                     ;DRIVE DID NOT ABORT SEARCH
 8810  041006  004737  044656          110$:  JSR     PC,TIMOUT               ;WAIT FOR GO TO RESET
 8811  041012  004737  044046                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 8812  041016  000404                        BR      120$    ;GO TO 120$ IF NO ERROR
 8813  041020  000240                        NOP                             ;RETURN HERE IF ERROR
 8814  041022  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
 8815  041024  000137  041130                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8816  041030                        120$:
 8817  041030  004737  045042                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
 8818  041034  000405                        BR      130$                    ;GO TO 130$ IF NO ERROR
 8819  041036  000240                        NOP                             ;RETURN HERE IF ERROR
 8820  041040  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8821  041042  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 8822  041044  000137  041130                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8823  041050                        130$:
 8824  041050  004737  060712                JSR     PC,STCDRVSTS            ;GO CHECK FOR CHANGES IN DRIVE STATUS
 8825  041054  000405                        BR      140$                    ;GO TO 140$ IF NO ERROR
 8826  041056  000240                        NOP                             ;RETURN HERE IF ERROR
 8827  041060  104000                        ERROR                           ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
 8828  041062  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 8829  041064  000137  041130                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8830  041070                        140$:
 8831  041070  004737  061416                JSR     PC,CMPERRSTS            ;CHECK ANY ERRORS NOT MASKED
 8832  041074  155760                        .WORD   ND!MSK!UNS              ;MASK FOR RMER1
 8833  041076  000010                        .WORD   DPE                     ;MASK FOR RMER2
```

```
                        T70     ABORT SEARCH TEST

  8834  041100  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
  8835  041102  000240                        NOP                     ;RETURN HERE IF ERROR
  8836  041104  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
  8837  041106  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
  8838  041110  000137  041130                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
  8839  041114                        150$:
  8840  041114  004737  045674                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
  8841  041120  000403                        BR      160$                    ;GO TO 160$ IF NO ERROR
  8842  041122  000240                        NOP                     ;RETURN HERE IF ERROR
  8843  041124  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
  8844  041126  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
  8845  041130                        160$:
  8846
  8847                                  ;;*****************************************************************
  8848                                  ;*TEST 71        SEARCH AT OFFSET
  8849                                  ;;*****************************************************************
  8850  041130                        TST71:
  8851  041130  000004                        SCOPE                   ;SCOPE CALL
  8852  041132  000240                        NOP                     ;START OF TEST
  8853  041134  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
  8854  041140  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
  8855  041144  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
  8856  041150  012737  000071  001226        MOV     #71,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  8857  041156  004737  043046                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
  8858  041162  054130                        .WORD   054130  ;TASK DESCRIPTOR
  8859  041164  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
  8860  041166  000240                        NOP                     ;RETURN HERE IF ERROR
  8861  041170  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
  8862  041172  000137  041472                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  8863  041176  012737  000000  001404  80$:  MOV     #0,RMDA0        ;TRACK=SECTOR=0
  8864  041204  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER=0
  8865  041212  012737  010000  001430        MOV     #FMT16,RMOF0    ;16 BIT FORMAT
  8866  041220  012737  000015  001376        MOV     #OFFSET!GO,RMCS10       ;OFFSET COMMAND
  8867  041226  012702  001533                MOV     #PUTINX,R2      ;SETUP REGISTER INDEX TABLE
  8868  041232  112722  000006                MOVB    #RMDA,(R2)+     ;FOR SEEK TO CYLINDER 0
  8869  041236  112722  000034                MOVB    #RMDC,(R2)+
  8870  041242  112722  000032                MOVB    #RMOF,(R2)+
  8871  041246  112722  000000                MOVB    #RMCS1,(R2)+
  8872  041252  112722  000200                MOVB    #200,(R2)+
  8873  041256  004737  044316                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
  8874  041262  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
  8875  041264  000240                        NOP                     ;RETURN HERE IF ERROR
  8876  041266  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
  8877  041270  000137  041472                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
  8878  041274  004737  043762        90$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
  8879  041300  004737  044656                JSR     PC,TIMOUT
  8880  041304  112737  000000  001533  120$:  MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
  8881  041312  112737  000200  001534        MOVB    #200,PUTINX+1
  8882  041320  012737  000031  001376        MOV     #SEARCH!GO,RMCS10;STORE SEARCH COMMAND
  8883  041326  004737  044316                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
  8884  041332  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
  8885  041334  000240                        NOP                     ;RETURN HERE IF ERROR
  8886  041336  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
  8887  041340  000137  041472                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
  8888  041344                        130$:
  8889  041344  004737  044656                JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
```

```
 8890  041350  004737  044046          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8891  041354  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
 8892  041356  000240                  NOP                     ;RETURN HERE IF ERROR
 8893  041360  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
 8894  041362  000137  041472          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 8895  041366                  140$:
 8896  041366  004737  045042          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8897  041372  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
 8898  041374  000240                  NOP                     ;RETURN HERE IF ERROR
 8899  041376  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8900  041400  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8901  041402  000137  041472          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8902  041406                  150$:
 8903  041406  004737  057346          JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
 8904  041412  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
 8905  041414  000240                  NOP                     ;RETURN HERE IF ERROR
 8906  041416  104000                  ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
 8907  041420  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8908  041422  000137  041472          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8909  041426                  160$:
 8910  041426  004737  061416          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 8911  041432  115760                  .WORD   NOTMSK          ;MASK FOR RMER1
 8912  041434  000010                  .WORD   DPE             ;MASK FOR RMER2
 8913  041436  000405                  BR      170$                    ;GO TO 170$ IF NO ERROR
 8914  041440  000240                  NOP                     ;RETURN HERE IF ERROR
 8915  041442  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 8916  041444  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8917  041446  000137  041472          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8918  041452                  170$:
 8919  041452  004737  045674          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8920  041456  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
 8921  041460  000240                  NOP                     ;RETURN HERE IF ERROR
 8922  041462  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8923  041464  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8924  041466  000137  041472          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 8925  041472                  180$:
 8926  041472                  190$:
 8927
 8928  041472  000137  042000          JMP     $EOSP           ;SKIP OVER HEAD ALIGNMENT SEEK
 8929
 8930                          ;PUT NEWTEST HERE
 8931                          ;;*****************************************************************
 8932                          ;*TEST 72        HEAD ALIGNMENT SEEK
 8933                          ;;*****************************************************************
 8934  041476                  TST72:
 8935  041476  000004                  SCOPE                   ;SCOPE CALL
 8936  041500  000240                  NOP                     ;START OF TEST
 8937  041502  012737  000001  001206  MOV     #1,$TIMES       ;LOAD ITERATION COUNT
 8938  041510  012737  041524  001122  MOV     #1$,$LPADR
 8939  041516  012737  041524  001124  MOV     #1$,$LPERR
 8940  041524                  1$:
 8941  041524  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 8942  041530  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 8943  041534  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 8944  041540  012737  000072  001226  MOV     #72,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8945
```

```
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;HEAD ALIGNMENT FOR THE RM03 DRIVE IS PERFORMED AT CYLINDER 245
        ;DECIMAL USING THE CE PACK. A 60 MINUTE WARMUP IS REQUIRED IF
        ;THE PACK WAS NOT IN THE DRIVE PRIOR TO HEAD ALIGNMENT OPERATION.
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

8957  041546                          10$:
8958  041546  004737  043046                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8959  041552  054130                          .WORD   054130  ;TASK DESCRIPTOR
8960  041554  000404                          BR      80$             ;GO TO 80$ IF NO ERROR
8961  041556  000240                          NOP                     ;RETURN HERE IF ERROR
8962  041560  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8963  041562  000137  042000                  JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
8964  041566                          80$:
8965  041566  012737  000000  001404          MOV     #0,RMDAO        ;TRACK = SECTOR = 0
8966  041574  012737  000365  001432          MOV     #245.,RMDCO     ;CYLINDER = 245
8967  041602  012737  000005  001376          MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
8968  041610  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
8969  041614  112722  000034                  MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
8970  041620  112722  000006                  MOVB    #RMDA,(R2)+
8971  041624  112722  000000                  MOVB    #RMCS1,(R2)+
8972  041630  112722  000200                  MOVB    #200,(R2)+      ;WRITE TERMINATOR
8973  041634  004737  044316                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8974  041640  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
8975  041642  000240                          NOP                     ;RETURN HERE IF ERROR
8976  041644  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
8977  041646  000137  042000                  JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8978  041652  004737  043762          90$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
8979  041656  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
8980  041662  004737  044046                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8981  041666  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
8982  041670  000240                          NOP                     ;RETURN HERE IF ERROR
8983  041672  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
8984  041674  000137  042000                  JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8985  041700                          100$:
8986  041700  004737  045042                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8987  041704  000405                          BR      110$            ;GO TO 110$ IF NO ERROR
8988  041706  000240                          NOP                     ;RETURN HERE IF ERROR
8989  041710  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8990  041712  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8991  041714  000137  042000                  JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
8992  041720                          110$:
8993  041720  004737  051750                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
8994  041724  000405                          BR      120$            ;GO TO 120$ IF NO ERROR
8995  041726  000240                          NOP                     ;RETURN HERE IF ERROR
8996  041730  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
8997  041732  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8998  041734  000137  042000                  JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
8999  041740                          120$:
9000  041740  004737  061416                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
9001  041744  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
```

```
9002  041746  000010                          .WORD   DPE             ;MASK FOR RMER2
9003  041750  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
9004  041752  000240                          NOP                     ;RETURN HERE IF ERROR
9005  041754  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
9006  041756  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9007  041760  000137  042000                  JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
9008  041764                          130$:
9009  041764  004737  045674                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
9010  041770  000403                          BR      140$                    ;GO TO 140$ IF NO ERROR
9011  041772  000240                          NOP                     ;RETURN HERE IF ERROR
9012  041774  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
9013  041776  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9014  042000                          140$:
```

```
                                    $EOSF:
042000                                      NOP
042000  000240                              NOP
042002  013700  001446                      MOV     TSTQUE,R0
042006  062700  000002                      ADD     #2,R0
042012  010037  001446                      MOV     R0,TSTQUE
042016  005710                              TST     (R0)
042020  001402                              BEQ     1$
042022  000137  006752                      JMP     READY
042026  012737  001450  001446    1$:       MOV     #TSTQUE+2,TSTQUE
                                    .SBTTL  END OF PASS ROUTINE


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

042034                              $EOP:
042034  000240                              NOP
042036  005037  001116                      CLR     $TSTNM          ;;ZERO THE TEST NUMBER
042042  005037  001206                      CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
042046  005237  001230                      INC     $PASS           ;;INCREMENT THE PASS NUMBER
042052  042737  100000  001230              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
042060  005327                              DEC     (PC)+           ;;LOOP?
042062  000001                    $EOPCT:   .WORD   1
042064  003063                              BGT     $DOAGN          ;;YES
042066  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
042070  000001                    $ENDCT:   .WORD   1
042072  042062                              $EOPCT
042074  104401  042102                      TYPE    ,65$            ;;TYPE ASCIZ STRING
042100  000407                              BR      64$             ;;GET OVER THE ASCIZ
                                    ;;65$:    .ASCIZ  <12><15>/END PASS #/
042120                              64$:
042120  013746  001230                      MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                    ;;TYPE PASS NUMBER
042124  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
042126  104401  042134                      TYPE    ,67$            ;;TYPE ASCIZ STRING
042132  000421                              BR      66$             ;;GET OVER THE ASCIZ
                                    ;;67$:    .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
042176                              66$:
042176  013746  001126                      MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                    ;;TOTAL NUMBER OF ERRORS
042202  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
042204  104401  001217                      TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
042210  005037  001126                      CLR     $ERTTL          ;;CLEAR ERROR TOTAL
042214  013700  000042            $GET42:   MOV     @#42,R0         ;;GET MONITOR ADDRESS
042220  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
042222  000005                              RESET                   ;;CLEAR THE WORLD
042224  004710                    $ENDAD:   JSR     PC,(R0)         ;;GO TO MONITOR
042226  000240                              NOP                     ;;SAVE ROOM
042230  000240                              NOP                     ;;FOR
042232  000240                              NOP                     ;;ACT11
042234                              $DOAGN:
042234  000137                              JMP     @(PC)+          ;;RETURN
042236  006752                    $RTNAD:   .WORD   READY
```

```
 9071  042240      377      377      000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 9072            042244                           .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.

042244                                  ERRTYP:
042244  104414                                  SAVREG
042246  032777  020000  136700                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
042254  001402                                  BEQ     1$              ;NO!!
042256  000137  042774                          JMP     21$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
042262  104401  001217                1$:       TYPE    ,$CRLF
042266  104401  043010                          TYPE    ,ERTY00         ;TYPE "UNT#"
042272  013746  001234                          MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
042276  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042300     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042301     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042302  005037  043000                          CLR     TSTNMB          ;LOAD TEST NUMBER FOR
042306  013737  001226  043000                  MOV     $TESTN,TSTNMB
042314  104401  043016                          TYPE    ,ERTY01         ;TYPE "TST#"
042320  013746  043000                          MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
042324  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042326     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042327     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042330  005037  043002                          CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
042334  113737  001130  043002                  MOVB    $ITEMB,ERRNMB   ;TYPEOUT
042342  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
042344  104401  043026                          TYPE    ,ERTY02         ;TYPE "ERR#"
042350  013746  043002                          MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
042354  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042356     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042357     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042360  104401  043035                2$:       TYPE    ,ERTY03         ;TYPE "PC="
042364  013746  001132                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
042370  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042372     006                                  .BYTE   6               ;;TYPE 6 DIGIT(S)
042373     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
042374  005737  043002                3$:       TST     ERRNMB          ;WAS AN ERROR CALLED?
042400  001575                                  BEQ     21$             ;NO!!
042402  104401  001217                          TYPE    ,$CRLF          ;YES-TYPE CRLF
042406  105037  043006                          CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
 9129  042412  105037  043007                CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
 9130  042416  013700  043002                MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
 9131  042422  006300                        ASL     R0              ;FOUR ENTRIES IN ERROR
 9132  042424  006300                        ASL     R0              ;TABLE
 9133  042426  006300                        ASL     R0
 9134  042430  062700  001552                ADD     #$ERRTB-8.,R0
 9135  042434  011001                        MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
 9136                                                                ;TABLE
 9137  042436  001507                        BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
 9138                                ;TYPE THE ERROR MESSAGE
 9139  042440  012102                4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
 9140  042442  001505                        BEQ     12$             ;BRANCH IF END OF MESSAGE
 9141  042444  010237  042612                MOV     R2,11$          ;LOAD ADDRESS OF STRING
 9142  042450  005037  043004                CLR     BOTADR          ;CLEAR BOT ADDRESS
 9143  042454  112203                5$:     MOVB    (R2)+,R3        ;END OF STRING??
 9144  042456  001454                        BEQ     10$             ;YES!!
 9145  042460  122703  000015                CMPB    #CR,R3          ;CARRIAGE RETURN??
 9146  042464  001003                        BNE     6$              ;NO!!
 9147  042466  105037  043007                CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
 9148  042472  000770                        BR      5$              ;GET NEXT CHARACTER
 9149  042474  122703  000012                CMPB    #LF,R3          ;LINE FEED??
 9150  042500  001765                        BEQ     5$              ;YES-GET NEXT CHARACTER
 9151  042502  122703  000011                CMPB    #HT,R3          ;HORIZONTAL TAB??
 9152  042506  001007                        BNE     8$              ;NO!!
 9153  042510  105237  043007          7$:   INCB    CHRCNT          ;ADJUST CHARACTER COUNT
 9154  042514  132737  000007  043007        BITB    #7,CHRCNT
 9155  042522  001372                        BNE     7$
 9156  042524  000407                        BR      9$
 9157  042526  105237  043007          8$:   INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
 9158  042532  122703  000040                CMPB    #' ,R3          ;SPACE??
 9159  042536  001002                        BNE     9$              ;NO!!
 9160  042540  010237  043004                MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
 9161  042544  122737  000100  043007  9$:   CMPB    #64.,CHRCNT     ;END OF LINE??
 9162  042552  103340                        BHIS    5$              ;NO!!
 9163  042554  013704  043004                MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
 9164  042560  001007                        BNE     90$             ;BRANCH IF SPACE DETECTED
 9165  042562  104401  001217                TYPE    ,$CRLF          ;TYPE CRLF
 9166  042566  105037  043007                CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9167  042572  013702  042612                MOV     11$,R2          ;SET UP R2 FOR TESTING
 9168  042576  000726                        BR      5$
 9169  042600  105044                90$:    CLRB    -(R4)           ;REPLACE SPACE
 9170  042602  112737  177777  043006        MOVB    #-1,BOTFLG      ;SET BOT FLAG
 9171  042610  104401                10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
 9172  042612  000000                11$:    .WORD                   ;STRING ADDRESS GOES HERE
 9173  042614  105737  043006                TSTB    BOTFLG          ;WAS STRING TRUNCATED??
 9174  042620  001707                        BEQ     4$              ;NO!!
 9175  042622  104401  001217                TYPE    ,$CRLF          ;YES-TYPE CRLF
 9176  042626  105037  043006                CLRB    BOTFLG          ;CLEAR BOT FLAG
 9177  042632  105037  043007                CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9178  042636  013702  043004                MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
 9179  042642  010237  042612                MOV     R2,11$          ;SETUP 11$ FOR TYPING
 9180  042646  112742  000040                MOVB    #' ,-(R2)       ;RESTORE SPACE
 9181  042652  105722                        TSTB    (R2)+           ;RESTORE R2
 9182  042654  000677                        BR      5$              ;TYPE REST OF STRING
 9183  042656                          12$:
 9184                                ;TYPE ERROR HEADER AND ERROR DATA
```

```
 9185  042656                          13$:
 9186  042656  016001  000002                MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
 9187  042662  001444                        BEQ     21$             ;BRANCH IF NO HEADER
 9188  042664  104401  001217                TYPE    ,$CRLF          ;(ASSUME NO DATA)
 9189  042670  016002  000004                MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
 9190  042674  016003  000006                MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
 9191  042700  012137  042710        14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
 9192  042704  001433                        BEQ     21$             ;BRANCH IF END OF HEADERS
 9193                                                                ;(ASSUME END OF DATA)
 9194  042706  104401                        TYPE
 9195  042710  000000        15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
 9196  042712  104401  001217                TYPE    ,$CRLF
 9197  042716  005702                        TST     R2              ;DATA WITH HEADER??
 9198  042720  001767                        BEQ     14$             ;NO!!
 9199  042722  012204                        MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
 9200  042724  012305                        MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
 9201  042726  105725        16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
 9202  042730  100407                        BMI     18$             ;BINARY
 9203  042732  001403                        BEQ     17$             ;OCTAL
 9204  042734  013446                        MOV     @(R4)+,-(SP)    ;DECIMAL
 9205  042736  104405                        TYPDS
 9206  042740  000405                        BR      19$
 9207  042742  013446        17$:    MOV     @(R4)+,-(SP)
 9208  042744  104402                        TYPOC
 9209  042746  000402                        BR      19$
 9210  042750  013446        18$:    MOV     @(R4)+,-(SP)
 9211  042752  104406                        TYPBN
 9212  042754  005714        19$:    TST     (R4)            ;MORE DATA??
 9213  042756  001403                        BEQ     20$             ;NO!!
 9214  042760  104401  043043                TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
 9215  042764  000760                        BR      16$             ;AND CONTINUE
 9216  042766  104401  001217        20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
 9217  042772  000742                        BR      14$             ;BEFORE NEXT HEADER
 9218  042774  104415        21$:    RESREG
 9219  042776  000207                        RTS     PC
 9220
 9221  043000  000000        TSTNMB: .WORD   0               ;TEST NUMBER
 9222  043002  000000        ERRNMB: .WORD   0               ;ERROR NUMBER
 9223  043004  000000        BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
 9224  043006     000        BOTFLG: .BYTE   0               ;BOT FLAG
 9225  043007     000        CHRCNT: .BYTE   0               ;CHARACTER COUNT
 9226
 9227  043010  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
 9228  043016  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
 9229  043024  000043
 9230  043026  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
 9231  043034     000
 9232  043035     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
 9233  043042     000
 9234  043043     040  000040          ERTY04: .ASCIZ  @ @
 9235                                  .EVEN
 9236
```

```
                                .SBTTL  TEST PREPARATION MODULE

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

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

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

043046                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
043046  017637  000000  043760          MOV     @(SP),500$      ;STORE DESCRIPTOR
043054  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
043060  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
043064  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
043070  032737  100000  043760          BIT     #BIT15,500$     ;SELECT DEVICE??
043076  001414                          BEQ     30$             ;NO!!
043100  004737  051536                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
043104  000411                          BR      30$             ;NO ERROR - CONTINUE
043106  000401                          BR      20$
043110  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
043112  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
043116  113776  043110  000000          MOVB    10$,@(SP)
043124  000137  043746                  JMP     400$
043130                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
043130  032737  040000  043760          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
043136  001453                          BEQ     120$            ;NO!!
```

```
9293  043140  004737  053210            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
9294  043144  000411                    BR      60$             ;CONTINUE - NO ERROR
9295  043146  000401                    BR      50$
9296  043150  000000            40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
9297  043152  062716  000004    50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9298  043156  113776  043150  000000    MOVB    40$,@(SP)
9299  043164  000137  043746            JMP     400$
9300  043170                    60$:
9301                              ;;*****************************************************************
9302                              ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
9303  043170  032737  000100  043760    BIT     #BIT6,500$      ;VERIFY??
9304  043176  001433                    BEQ     120$            ;NO!!
9305  043200  004737  043762            JSR     PC,GETSTS       ;SETUP INDEX TABLE
9306  043204  004737  044046            JSR     PC,GET          ;GO GET STATUS
9307  043210  000411                    BR      90$             ;NO ERROR GETTING STATUS
9308  043212  000401                    BR      80$
9309  043214  000000            70$:    .WORD                   ;ERROR FROM GETTING STATUS
9310  043216  062716  000004    80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9311  043222  113776  043214  000000    MOVB    70$,@(SP)
9312  043230  000137  043746            JMP     400$
9313  043234  004737  053326    90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
9314  043240  000412                    BR      120$            ;NO ERROR IN CLEAR
9315  043242  000401                    BR      110$
9316  043244  000000           100$:    .WORD                   ;ERROR IN STATUS CLEAR
9317  043246  005726           110$:    TST     (SP)+           ;STRIP RETURN ADDRESS TO
9318  043250  062716  000004            ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
9319  043254  113776  043244  000000    MOVB    100$,@(SP)      ;ERROR TO USER
9320  043262  000137  043746            JMP     400$
9321  043266                   120$:
9322
9323                              ;;*****************************************************************
9324                              ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
9325                              ;NOT VALID
9326  043266  032737  010000  043760    BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
9327  043274  001511                    BEQ     240$            ;NO!!
9328  043276  112737  000012  001504    MOVB    #RMDS,GETINX    ;GET RMDS
9329  043304  112737  000200  001505    MOVB    #200,GETINX+1
9330  043312  004737  044046            JSR     PC,GET
9331  043316  000411                    BR      150$            ;NO ERROR GETTING RMDS
9332  043320  000401                    BR      140$
9333  043322  000000           130$:    .WORD
9334  043324  062716  000004   140$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9335  043330  113776  043322  000000    MOVB    130$,@(SP)
9336  043336  000137  043746            JMP     400$
9337  043342  032737  000100  001340   150$:    BIT     #VV,RMDSI       ;IS VOLUME VALID??
9338  043350  001063                    BNE     240$            ;YES!!
9339  043352  005037  001472            CLR     MEDENB          ;CLEAR MEDIA ENABLE
9340  043356  012737  000023  001376    MOV     #PAKACK!GO,RMCS10       ;LOAD PACK ACK COMMAND
9341  043364  112737  000000  001533    MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
9342  043372  112737  000200  001534    MOVB    #200,PUTINX+1
9343  043400  004737  044316            JSR     PC,PUT          ;GO WRITE COMMAND
9344  043404  000410                    BR      180$            ;NO ERROR LOADING REGISTER
9345  043406  000401                    BR      170$
9346  043410  000000           160$:    .WORD                   ;ERROR FROM PUT SUB
9347  043412  062716  000004   170$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9348  043416  113776  043410  000000    MOVB    160$,@(SP)
```

```
 9349  043424  000550                          BR      400$
 9350  043426                          180$:
 9351
 9352                                  ;;*****************************************************************
 9353                                  ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
 9354  043426  032737  000020  043760          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
 9355  043434  001431                          BEQ     240$            ;NO!!
 9356  043436  004737  043762                  JSR     PC,GETSTS       ;SETUP FOR STATUS
 9357  043442  004737  044046                  JSR     PC,GET          ;GO GET STATUS
 9358  043446  000410                          BR      210$            ;NO ERROR GETTING STATUS
 9359  043450  000401                          BR      200$
 9360  043452  000000                  190$:   .WORD                   ;ERROR FROM GET SUB
 9361  043454  062716  000004          200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9362  043460  113776  043452  000000          MOVB    190$,@(SP)
 9363  043466  000527                          BR      400$
 9364  043470  004737  054206          210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
 9365  043474  000411                          BR      240$            ;NO ERROR
 9366  043476  000401                          BR      230$
 9367  043500  000000                  220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
 9368  043502  005726                  230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
 9369  043504  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9370  043510  113776  043500  000000          MOVB    220$,@(SP)
 9371  043516  000513                          BR      400$
 9372  043520                          240$:
 9373
 9374                                  ;;*****************************************************************
 9375                                  ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
 9376  043520  032737  004000  043760          BIT     #BIT11,500$     ;RECALIBRATE??
 9377  043526  001513                          BEQ     410$            ;NO!!
 9378  043530  112737  000012  001504          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
 9379  043536  112737  000200  001505          MOVB    #200,GETINX+1
 9380  043544  004737  044046                  JSR     PC,GET          ;GO GET RMDS
 9381  043550  000410                          BR      270$            ;NO ERROR GETTING RMDS
 9382  043552  000401                          BR      260$
 9383  043554  000000                  250$:   .WORD                   ;ERROR FROM GET SUB
 9384  043556  062716  000004          260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9385  043562  113776  043554  000000          MOVB    250$,@(SP)
 9386  043570  000466                          BR      400$
 9387  043572  032737  020000  001340  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
 9388  043600  001466                          BEQ     410$            ;NO!!
 9389  043602  012737  000007  001376          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
 9390  043610  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
 9391  043616  112737  000200  001534          MOVB    #200,PUTINX+1
 9392  043624  004737  044316                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
 9393  043630  000410                          BR      300$            ;NO ERROR
 9394  043632  000401                          BR      290$
 9395  043634  000000                  280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
 9396  043636  062716  000004          290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9397  043642  113776  043634  000000          MOVB    280$,@(SP)
 9398  043650  000436                          BR      400$
 9399  043652  004737  043762          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
 9400  043656  004737  044656                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
 9401
 9402                                  ;;*****************************************************************
 9403                                  ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
 9404  043662  032737  000010  043760          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 9405  043670  001432                    BEQ     410$            ;NO!!
 9406  043672  004737  044046            JSR     PC,GET          ;GO GET STATUS
 9407  043676  000410                    BR      330$            ;NO ERROR GETTING STATUS
 9408  043700  000401                    BR      320$
 9409  043702  000000            310$:   .WORD                   ;ERROR FROM GET
 9410  043704  062716  000004    320$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9411  043710  113776  043702  000000    MOVB    310$,@(SP)
 9412  043716  000413                    BR      400$
 9413  043720  004737  055002    330$:   JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 9414  043724  000414                    BR      410$            ;NO ERROR DURING RECALIBRATE
 9415  043726  000401                    BR      350$
 9416  043730  000000            340$:   .WORD                   ;ERROR DURING RECALIBRATE
 9417  043732  005726            350$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
 9418  043734  062716  000004            ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9419  043740  113776  043730  000000    MOVB    340$,@(SP)
 9420  043746  162716  000002    400$:   SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 9421  043752  000240                    NOP
 9422  043754  000240                    NOP
 9423  043756  000207            410$:   RTS     PC              ;RETURN TO USER
 9424
 9425  043760  000000            500$:   .WORD                   ;TASK/VERIFY DESCRIPTOR
```

```
                                        .SBTTL  GET STATUS SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
                                        ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
                                        ;AND THEN RETURNS TO THE USER.
                                        ;
                                        ;CALL:  JSR     PC,GETSTS
                                        ;       ???                     RETURN HERE
                                        ;
043762                                  GETSTS:
043762  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043764  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043766  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043770  012700  001504                          MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
043774  012701  001376                          MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
044000  012702  000046                          MOV     #RMEC2,R2       ;R2 = REGISTER INDE
044004  110220                        2$:       MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
044006  005041                                  CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
044010  162702  000002                3$:       SUB     #2,R2           ;DECREMENT TO NEXT INDEX
044014  100405                                  BMI     4$              ;BRANCH OUT IF DONE
044016  022702  000022                          CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
044022  001370                                  BNE     2$
044024  005041                                  CLR     -(R1)
044026  000770                                  BR      3$
044030  112720  000200                4$:       MOVB    #200,(R0)+      ;WRITE TERMINATOR
044034  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
044036  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
044040  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
044042  000240                                  NOP
044044  000207                                  RTS     PC
```

```
                                .SBTTL  GET SUBROUTINE

                                ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
                                ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
                                ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
                                ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
                                ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
                                ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
                                ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
                                ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
                                ;WHICH SHOULD FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;(1)     "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
                                ;        VALUES AND TERMINATED WITH A CONTROL BYTE
                                ;(2)     "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
                                ;        UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
                                ;        TO REGISTERS NOT READ, ARE NOT CHANGED.)
                                ;(3)     JSR     PC,GET
                                ;        BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;        NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;        ERROR                   SUB DEFINES ERROR NUMBER
                                ;        ???
                                ;
044046  000240          GET:    NOP
044050  062716  000004          ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
044054  105076  000000          CLRB    @(SP)           ;ERROR CALL
044060  162716  000004          SUB     #4,(SP)
044064  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044066  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044070  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044072  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044074  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
044076  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044102  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044106  013700  001276          MOV     $BASE,R0
044112  012702  001326          MOV     #GETBUF,R2
044116  012704  001504          MOV     #GETINX,R4
044122  012737  044230  000004  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
044130  012737  000300  000006  MOV     #PR6,ERRVEC+2
044136  016037  000010  001174  1$:    MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
044144  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
044152  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
044160  001007                  BNE     3$              ;YES!!
044162  062766  000004  000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
044170  112776  000112  000016  MOVB    #112,@16(SP)    ;ERROR CALL
044176  000423                  BR      7$
044200  105714          3$:     TSTB    (R4)            ;DONE??
044202  100433                  BMI     9$              ;YES!!
044204  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
044206  042701  177700          BIC     #^C<IDXMSK>,R1  ;CLEAR ANY SIGN EXTENSION
044212  060001                  ADD     R0,R1
044214  112403                  MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
044216  042703  177700          BIC     #^C<IDXMSK>,R3  ;CLEAR ANY SIGN EXTENSION
044222  060203                  ADD     R2,R3
044224  011113                  MOV     (R1),(R3)       ;READ REGISTER
```

```
 9513  044226  000764                          BR      3$
 9514
 9515  044230  022626                  5$:     CMP     (SP)+ (SP)+     ;RESTORE STACK
 9516  044232  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
 9517  044240  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
 9518  044246  162766  000002  000016  7$:     SUB     #2,16(SP)
 9519  044254  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
 9520  044256  100405                          BMI     9$              ;YES!!
 9521  044260  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
 9522  044262  112403                          MOVB    (R4)+,R3        ;LOCATIONS
 9523  044264  060203                          ADD     R2,R3
 9524  044266  005013                          CLR     (R3)
 9525  044270  000771                          BR      8$
 9526  044272                          9$:
 9527  044272  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 9528  044276  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 9529  044302  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
 9530  044304  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 9531  044306  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 9532  044310  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9533  044312  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9534  044314  000207                          RTS     PC              ;RETURN
 9535
```

```
                                .SBTTL  PUT SUBROUTINE

                                ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
                                ;"PUT INDEX TABLE" WITH THE CONTENTS OF THE CORRESPONDIN^
                                ;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
                                ;       ???

                                PUT:    NOP
 044316  000240                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 044320  010046                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 044322  010146                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 044324  010246                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 044326  010346                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 044330  010446                         MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 044332  013746  000004                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
 044336  013746  000006                 MOV     $BASE,R0
 044342  013700  001276                 MOV     #PUTBUF,R2
 044346  012702  001376                 MOV     #PUTINX,R4
 044352  012704  001533                 MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
 044356  012737  044464  000004         MOV     #PR6,ERRVEC+2
 044364  012737  000300  000006         MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
 044372  016037  000010  001174  1$:    MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
 044400  016037  000000  001176         BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
 044406  032737  004000  001176         BNE     3$              ;YES!!
 044414  001007                         ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
 044416  062766  000004  000016         MOVB    #112,@16(SP)    ;USER'S ERROR CALL
 044424  112776  000112  000016         BR      7$
 044432  000423                 3$:     TSTB    (R4)            ;DONE??
 044434  105714                         BMI     9$              ;YES!!
 044436  100424                         MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
 044440  111401                         BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
 044442  042701  177700                 ADD     R0,R1
 044446  060001                         MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
 044450  112403                         BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
 044452  042703  177700                 ADD     R2,R3
 044456  060203                         MOV     (R3),(R1)       ;WRITE REGISTER
 044460  011311                         BR      3$
 044462  000764
                                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
 044464  022626                         ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
 044466  062766  000004  000016         MOVB    #7,@16(SP)      ;USER'S ERROR CALL
 044474  112776  000007  000016         SUB     #2,16(SP)
 044502  162766  000002  000016  7$:
```

```
 9592
 9593  044510                          9$:
 9594  044510  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 9595  044514  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 9596  044520  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
 9597  044522  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 9598  044524  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 9599  044526  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 9600  044530  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9601  044532  000207                        RTS     PC              ;RETURN
 9602
```

```
                                .SBTTL  SIZE CLOCK SUBROUTINE

                                SIZCLK:
044534                                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044534  013746  000004                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044540  013746  000006                   MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
044544  012737  044602  000004           MOV     #PR6,ERRVEC+2
044552  012737  000300  000006           MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
044560  012737  177546  001500           MOV     #100,CLKVCT
044566  012737  000100  001502           TST     @CLKADR         ;TEST FOR KW11-L PRESENT
044574  005777  134700                   BR      3$              ;YES - KW11-L IS PRESENT
044600  000421                  1$:      CMP     (SP)+,(SP)+     ;RESTORE SP
044602  022626                           MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
044604  012737  044634  000004           MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
044612  012737  172540  001500           MOV     #104,CLKVCT
044620  012737  000104  001502           TST     @CLKADR         ;TEST FOR KW11-P PRESENT
044626  005777  134646                   BR      3$              ;YES - KW11-P IS PRESENT
044632  000404                  2$:      CMP     (SP)+,(SP)+     ;RESTORE SP
044634  022626                           ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
044636  062766  000002  000004   3$:
044644                                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
044644  012637  000006                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
044650  012637  000004                   RTS     PC              ;RETURN TO USER
044654  000207
```

```
                                        .SBTTL  TIMEOUT SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
                                        ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
                                        ;
                                        ;CALL:  JSR     PC,TIMOUT
                                        ;       ???                     RETURN HERE
                                        ;
044656                                  TIMOUT:
044656  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044660  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044662  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044664  013746  000004                         MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044670  013746  000006                         MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044674  012737  044776  000004                MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
044702  012737  000300  000006                MOV     #PR6,ERRVEC+2
044710  013700  001276                        MOV     $BASE,R0        ;R0=RM03 ADDRESS
044714  013701  001500                        MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
044720  012702  000037                        MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
044724  020127  172540          1$:           CMP     R1,#172540      ;KW11-P CLOCK??
044730  001003                                BNE     2$              ;NO!!
044732  012761  000001  000002                MOV     #1,2(R1)        ;SET COUNTER
044740  012711  000005          2$:           MOV     #BIT2!BIT0,(R1) ;START COUNTER
                                        ;
044744  016046  000000          3$:           MOV     RMCS1(R0),-(SP) ;GET STATUS
044750  042716  177576                        BIC     #^C<RDY!GO>,(SP)
044754  022726  000200                        CMP     #RDY,(SP)+      ;RDY=1,GO=0??
044760  001420                                BEQ     5$              ;YES!!
044762  032711  000200                        BIT     #BIT7,(R1)      ;TIMER DONE??
044766  001766                                BEQ     3$              ;NO!!
044770  005302                                DEC     R2              ;DEC NUMBER OF CYCLES
044772  001354                                BNE     1$              ;CONTINUE IF NOT DONE
044774  000412                                BR      5$
044776  022626                  4$:           CMP     (SP)+,(SP)+     ;ADJUST STACK
045000  062766  000004  000012                ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
045006  112776  000007  000012                MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
045014  162766  000002  000012                SUB     #2,12(SP)
                                        ;
045022                                  5$:
045022  012637  000006                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
045026  012637  000004                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
045032  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
045034  012601                                MOV     (SP)+,R1        ;;POP STACK INTO R1
045036  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
045040  000207                                RTS     PC              ;RETURN TO USER
                                        ;
```

```
                                .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
                                ;
                                ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
                                ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
                                ;FOLLOWING CHECKS ARE MADE:
                                ;
                                ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
                                ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
                                ;
                                ;       .SELECTED UNIT IS AVAILABLE, I.E., DVH (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.
                                ;
                                ;       .($UNIT) CONTAINS THE DRIVE NUMBER
                                ;
                                ;THE SUBROUTINE IS CALLED AS FOLLOWS:
                                ;
                                ;(1)    JSR     PC,PRIERR
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS
                                ;
045042                          PRIERR:
                                ;
                                ;CLEAR USER'S ERROR CALL
045042  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
045046  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
045052  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                                ;
                                ;
                                ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
045056  013737  001336  001142          MOV     RMCS2I,$BODAT   ;CORRECT UNIT SELECTED??
045064  042737  177770  001142          BIC     #^CUNTMSK,$BODAT
045072  013737  001234  001140          MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
045100  042737  177770  001140          BIC     #^CUNTMSK,$GDDAT
045106  123737  001140  001142          CMPB    $GDDAT,$BODAT   ;COMPARE EXPECTED AND RECEIVED
                                                                ;DRIVE NUMBERS
045114  001415                          BEQ     1$              ;YES!!
045116  062716  000004                  ADD     #4,(SP)
045122  112776  000001  000000          MOVB    #1,@(SP)        ;ERROR 1
```

```
 9728  045130  162716  000002                 SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9729  045134  004736                         JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
 9730  045136  162716  000010                 SUB     #10,(SP)         ;RESTORE (SP)
 9731  045142  000240                         NOP
 9732  045144  000137  045664                 JMP     10$              ;SKIP OTHER CHECKS
 9733  045150                             1$:
 9734
 9735                                     ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
 9736                                     ;THE DEVICE IS NONEXISTANT
 9737  045150  032737  004000  001326             BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
 9738  045156  001045                         BNE     5$               ;YES!!
 9739  045160  013737  001326  001140         MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9740  045166  052737  004000  001140         BIS     #DVA,$GDDAT
 9741  045174  013737  001326  001142         MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9742  045202  062716  000004                 ADD     #4,(SP)
 9743  045206  112776  000002  000000         MOVB    #2,@(SP)         ;ERROR #2
 9744  045214  032737  010000  001336         BIT     #NED,RMCS2I      ;WAS NED SET??
 9745  045222  001414                         BEQ     2$               ;NO!!
 9746  045224  013737  001336  001140         MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
 9747  045232  013737  001336  001142         MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
 9748  045240  042737  010000  001140         BIC     #NED,$GDDAT
 9749  045246  112776  000003  000000         MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
 9750  045254  162716  000002             2$: SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9751  045260  004736                         JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
 9752  045262  162716  000010                 SUB     #10,(SP)         ;RESTORE (SP)
 9753  045266  000240                         NOP
 9754  045270  000575                         BR      10$              ;SKIP OTHER CHECKS
 9755  045272                             5$:
 9756
 9757                                     ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
 9758  045272  032737  000200  001326         BIT     #RDY,RMCS1I      ;CONTROLLER READY??
 9759  045300  001030                         BNE     7$               ;YES!!
 9760  045302  013737  001326  001140         MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9761  045310  052737  000200  001140         BIS     #RDY,$GDDAT
 9762  045316  042737  160001  001140         BIC     #SC!TRE!MCPE!GO,$GDDAT
 9763  045324  013737  001326  001142         MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9764  045332  062716  000004                 ADD     #4,(SP)
 9765  045336  112776  000004  000000         MOVB    #4,@(SP)         ;ERROR #4
 9766  045344  162716  000002                 SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9767  045350  004736                         JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
 9768  045352  162716  000010                 SUB     #10,(SP)         ;RESTORE (SP)
 9769  045356  000240                         NOP
 9770  045360  000541                         BR      10$              ;SKIP OTHER CHECKS
 9771  045362                             7$:
 9772
 9773                                     ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
 9774  045362  032737  000001  001326         BIT     #GO,RMCS1I       ;GO RESET??
 9775  045370  001431                         BEQ     8$               ;YES!!
 9776  045372  032737  000200  001340         BIT     #DRY,RMDSI       ;DRIVE READY??
 9777  045400  001025                         BNE     8$               ;YES!!
 9778  045402  013737  001326  001140         MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9779  045410  042737  160001  001140         BIC     #SC!TRE!MCPE!GO,$GDDAT
 9780  045416  013737  001326  001142         MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9781  045424  062716  000004                 ADD     #4,(SP)
 9782  045430  112776  000005  000000         MOVB    #5,@(SP)         ;ERROR #5
 9783  045436  162716  000002                 SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
 9784  045442  004736                          JSR     ^PC,@(SP)+      ;REPORT DRIVE NOT READY
 9785  045444  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
 9786  045450  000240                          NOP
 9787  045452  000504                          BR      10$
 9788  045454                              8$:
 9789
 9790                                        ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
 9791                                        ;PARITY ON THE MASSBUS CONTROL BUS
 9792  045454  032737  020000  001326          BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
 9793  045462  001425                          BEQ     9$              ;NO!!
 9794  045464  013737  001326  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
 9795  045472  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
 9796  045500  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
 9797  045506  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
 9798  045512  112776  000013  000000          MOVB    #13,@(SP)       ;ERROR #47
 9799  045520  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9800  045524  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9801  045526  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
 9802  045532  000240                          NOP
 9803  045534  000453                          BR      10$
 9804  045536                              9$:
 9805
 9806                                        ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
 9807  045536  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
 9808  045544  001451                          BEQ     11$             ;NO!!
 9809  045546  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
 9810  045554  001045                          BNE     11$             ;NOT SURE!!
 9811  045556  032737  000010  001412          BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
 9812  045564  001413                          BEQ     93$             ;NO!!
 9813  045566  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9814  045570  012700  001533                  MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
 9815  045574  122710  000014          91$:    CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
 9816  045600  001002                          BNE     92$
 9817  045602  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9818  045604  000431                          BR      11$             ;PAR WAS SET BY TEST
 9819  045606  105720                  92$:    TSTB    (R0)+           ;END OF TABLE??
 9820  045610  100371                          BPL     91$             ;NO!!
 9821  045612  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9822  045614  013737  001342  001140  93$:    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 9823  045622  042737  000010  001140          BIC     #PAR,$GDDAT
 9824  045630  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 9825  045636  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 9826  045642  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
 9827  045650  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9828  045654  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
 9829  045656  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 9830  045662  000240                          NOP
 9831  045664  062716  000010          10$:    ADD     #10,(SP)                ;RETURN TO ERROR
 9832  045670  000240                  11$:    NOP                     ;RETURN TO NO ERROR
 9833  045672  000207                          RTS     PC
 9834
```

```
                                .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.

045674                          SECERR:

                                ;;*****************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
045674  013737  001376  051534          MOV     RMCS1O,S15$     ;STORE FUNCTION CODE
045702  042737  177701  051534          BIC     #^C<F0!F1!F2!F3!F4>,S15$
045710  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
045714  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
045720  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
045724  032737  000200  001340          BIT     #DRY,RMDSI      ;DRIVE READY??
045732  001024                  BNE     5$              ;YES!!
045734  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
045742  042737  177577  001142          BIC     #^CDRY,$BDDAT
045750  012737  000200  001140          MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
045756  062716  000004                  ADD     #4,(SP)
045762  112776  000010  000000          MOVB    #10,@(SP)       ;ERROR NUMBER
045770  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
045774  004736                  JSR     PC,@(SP)+       ;REPORT NOT READY
045776  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
046002  000240                  NOP

                                ;REPORT ERROR IF GO BIT IS NOT RESET
046004  032737  000001  001326  5$:     BIT     #GO,RMCS1I      ;GO BIT RESET??
046012  001423                  BEQ     10$             ;YES!!
046014  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
046022  042737  177776  001142          BIC     #^CGO,$BDDAT
046030  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
046034  062716  000004                  ADD     #4,(SP)
```

```
 9891  046040  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
 9892  046046  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9893  046052  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
 9894  046054  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9895  046060  000240                          NOP
 9896
 9897                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
 9898  046062  013737  001326  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
 9899  046070  042737  177701  001142          BIC     #^C76,$BDDAT
 9900  046076  013737  051534  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
 9901  046104  023737  001142  001140          CMP     $BDDAT,$GDDAT
 9902  046112  001413                          BEQ     15$             ;YES!!
 9903  046114  062716  000004                  ADD     #4,(SP)
 9904  046120  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
 9905  046126  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9906  046132  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
 9907  046134  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9908  046140  000240                          NOP
 9909  046142                          15$:
 9910                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
 9911                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
 9912                                  ;OTHER ERRORS ARE SET
 9913  046142  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
 9914  046146  005737  001342                  TST     RMER1I          ;IS RMER1 =0??
 9915  046152  001003                          BNE     20$             ;NO!!
 9916  046154  005737  001370                  TST     RMER2I          ;IS RMERZ=0??
 9917  046160  001403                          BEQ     25$             ;YES!!
 9918  046162  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
 9919  046170  013737  001340  001142  25$:     MOV     RMDSI,$BDDAT
 9920  046176  042737  137777  001142          BIC     #^CERR,$BDDAT
 9921  046204  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
 9922  046212  001412                          BEQ     30$             ;YES!!
 9923  046214  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9924  046220  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
 9925  046226  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9926  046232  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
 9927  046234  162716  000010                  SUB     #10,(SP)
 9928
 9929                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
 9930                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
 9931                                  ;SET TRE IS SET
 9932  046240  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
 9933  046244  013746  001336                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
 9934  046250  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
 9935  046254  001010                          BNE     35$             ;YES!!
 9936  046256  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
 9937  046264  001407                          BEQ     40$             ;NO!!
 9938  046266  022737  000030  051534          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9939  046274  103003                          BHIS    40$             ;NO!!
 9940  046276  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
 9941  046304  013737  001326  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
 9942  046312  042737  137777  001142          BIC     #^CTRE,$BDDAT
 9943  046320  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
 9944  046326  001413                          BEQ     45$             ;YES!!
 9945  046330  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9946  046334  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
 9947  046342  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9948  046346  004736                    JSR     PC,@(SP)+        ;REPORT TRE ERROR
 9949  046350  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
 9950  046354  000240                    NOP
 9951  046356                        45$:
 9952
 9953                                   ;;*****************************************************************
 9954                                   ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
 9955                                   ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
 9956                                   ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
 9957                                   ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
 9958                                   ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
 9959                                   ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
 9960
 9961                                   ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
 9962  046356  010046                    MOV     R0,-(SP)         ;;PUSH R0 ON STACK
 9963  046360  013700  051534            MOV     51$$,R0          ;R0 = FUNCTION CODE
 9964  046364  016037  066770  051526    MOV     FNCDTB(R0),500$  ;STORE ENTRY
 9965  046372  012600                    MOV     (SP)+,R0         ;;POP STACK INTO R0
 9966
 9967                                   ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
 9968                                   ;ATA IS NOT SET AND SHOULD BE SET.
 9969  046374  013737  051526  001140    MOV     50$$,$GDDAT      ;GET EXPECTED ATA STATUS
 9970  046402  032737  040000  001340    BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
 9971  046410  001403                    BEQ     50$              ;NO !!
 9972  046412  052737  100000  001140    BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
 9973  046420  042737  077777  001140  50$:    BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
 9974  046426  013737  001340  001142    MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
 9975  046434  042737  077777  001142    BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
 9976  046442  023737  001140  001142    CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
 9977  046450  001413                    BEQ     55$              ;YES !!
 9978  046452  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9979  046456  112776  000006  000000    MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
 9980  046464  162716  000002            SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9981  046470  004736                    JSR     PC,@(SP)+        ;REPORT ERROR
 9982  046472  162716  000010            SUB     #10,(SP)         ;RESTORE SP
 9983  046476  000240                    NOP
 9984  046500                        55$:
 9985
 9986                                   ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
 9987                                   ;WITH FUNCTION CODE TABLE
 9988  046500  013737  051526  001140    MOV     500$,$GDDAT      ;GET EXPECTED ILF
 9989  046506  042737  177776  001140    BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
 9990  046514  013737  001342  001142    MOV     RMERII,$BDDAT    ;GET RECEIVED ILF
 9991  046522  042737  177776  001142    BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
 9992  046530  023737  001140  001142    CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
 9993  046536  001412                    BEQ     60$              ;YES !!
 9994  046540  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9995  046544  112776  000254  000000    MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9996  046552  162716  000002            SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9997  046556  004736                    JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
 9998  046560  162716  000010            SUB     #10,(SP)         ;MOVE SP TO NO ERROR
 9999  046564  005037  001140        60$:    CLR     $GDDAT           ;CLEAR EXPECTED STATUS
10000
10001                                   ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
10002  046570  013746  051526            MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
10003  046574  052716  137777                        BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
10004  046600  013737  001336  001142                MOV     RMCS2I,$BODAT   ;RECEIVED STATUS
10005  046606  042637  001142                        BIC     (SP)+,$BODAT    ;CLEAR WCE IF ENABLED
10006  046612  001412                                BEQ     90$             ;BRANCH IF WCE OK
10007  046614  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10008  046620  112776  000026  000000                MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
10009  046626  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10010  046632  004736                                JSR     PC,@(SP)+       ;REPORT ERROR
10011  046634  162716  000010                        SUB     #10,(SP)        ;RESTORE ERROR
10012  046640                                90$:
10013
10014                                        ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
10015  046640  013746  051526                        MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
10016  046644  052716  157777                        BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
10017  046650  013737  001342  001142                MOV     RMER1I,$BODAT   ;GET RECEIVED STATUS
10018  046656  042637  001142                        BIC     (SP)+,$BODAT    ;CLEAR OPI IF ENABLED
10019  046662  001412                                BEQ     100$            ;BRANCH IF OPI OK
10020  046664  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10021  046670  112776  000164  000000                MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
10022  046676  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10023  046702  004736                                JSR     PC,@(SP)+       ;REPORT ERROR
10024  046704  162716  000010                        SUB     #10,(SP)        ;RESTORE SP
10025  046710                                100$:
10026
10027                                        ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
10028                                        ;SET AND VV IS NOT RESET
10029  046710  013746  051526                        MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
10030  046714  032737  000100  001340                BIT     #VV,RMDSI       ;IS VV SET
10031  046722  001402                                BEQ     105$            ;NO !!
10032  046724  042716  010000                        BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
10033  046730  052716  167777                105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
10034  046734  013737  001370  001142                MOV     RMER2I,$BODAT   ;GET RECEIVED STATUS
10035  046742  042637  001142                        BIC     (SP)+,$BODAT    ;CLEAR IVC IF ENABLED
10036  046746  001412                                BEQ     110$            ;BRANCH IF IVC OK
10037  046750  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10038  046754  112776  000165  000000                MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
10039  046762  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10040  046766  004736                                JSR     PC,@(SP)+       ;REPORT ERROR
10041  046770  162716  000010                        SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10042  046774                                110$:
10043
10044                                        ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10045                                        ; ALL WRITE ERRORS, I.E.,
10046                                        ;       RMER1 - WLE, WCF
10047                                        ;       RMER2 - DPE
10048                                        ;       RMCS2 - UPE.
10049                                        ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
10050                                        ;WRITE ERROR ENABLE BIT IS RESET.
10051
10052                                        ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
10053                                        ;THE DRIVE IS NOT WRITE PROTECTED
10054  046774  012746  177777                        MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10055  047000  032737  004000  051526                BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10056  047006  001404                                BEQ     115$            ;NO !!
10057  047010  032737  004000  001340                BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
10058  047016  001002                                BNE     120$            ;YES !!
```

```
10059  047020  042716  004000          115$:   BIC     #WLE,(SP)       ;RESET WLE ENABLE
10060  047024  013737  001342  001142  120$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10061  047032  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WLE IF ENABLED
10062  047036  001412                          BEQ     125$            ;BRANCH IF WLE OK
10063  047040  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10064  047044  112776  000023  000000          MOVB    #23,@(SP)       ;WRITE ERROR NUMBER IN CALL
10065  047052  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10066  047056  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10067  047060  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10068  047064                          125$:
10069
10070                                  ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
10071  047064  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10072  047070  032737  004000  051526          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10073  047076  001002                          BNE     130$            ;YES !!
10074  047100  042716  000040                  BIC     #WCF,(SP)       ;DISABLE WCF ERROR
10075  047104  013737  001342  001142  130$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10076  047112  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET WCF IF ENABLED
10077  047116  001412                          BEQ     135$            ;BRANCH IF WCF OK
10078  047120  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CAL_
10079  047124  112776  000025  000000          MOVB    #25,@(SP)       ;WRITE ERROR NUMBER IN CAI _
10080  047132  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10081  047136  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10082  047140  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10083  047144                          135$:
10084
10085                                  ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENAPLED
10086  047144  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
10087  047150  032737  004000  051526          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10088  047156  001002                          BNE     140$            ;YES !!
10089  047160  042716  000010                  BIC     #DPE,(SP)       ;RESET DPE ENABLE
10090  047164  013737  001370  001142  140$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10091  047172  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET DPE IF ENABLED
10092  047176  001412                          BEQ     145$            ;BRANCH IF DPE OK
10093  047200  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10094  047204  112776  000040  000000          MOVB    #40,@(SP)       ;WRITE ERROR NUMBER IN CALL
10095  047212  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10096  047216  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10097  047220  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10098  047224                          145$:
10099
10100                                  ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10101  047224  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
10102  047230  032737  004000  051526          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10103  047236  001002                          BNE     150$            ;YES !!
10104  047240  042716  020000                  BIC     #UPE,(SP)       ;DISABLE UPE ERROR
10105  047244  013737  001336  001142  150$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10106  047252  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET UPE IF ENABLED
10107  047256  001412                          BEQ     155$            ;BRANCH IF UPE OK
10108  047260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10109  047264  112776  000024  000000          MOVB    #24,@(SP)       ;WRITE ERROR NUMBER IN CALL
10110  047272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10111  047276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10112  047300  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10113  047304                          155$:
10114
```

```
10115                                          ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
10116  047304  013746  051526                          MOV     S00$,-(SP)      ;GET IAE ENABLE
10117  047310  052716  175777                          BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
10118  047314  013737  001342  001142                  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10119  047322  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
10120  047326  001412                                  BEQ     160$            ;BRANCH IF IAE IS OK
10121  047330  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10122  047334  112776  000166  000000                  MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
10123  047342  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10124  047346  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10125  047350  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10126  047354                                 160$:
10127
10128                                          ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10129                                          ; ALL READ/WRITE ERRORS, I.E.,
10130                                          ;       RMCS1 - TRE
10131                                          ;       RMCS2 - DLT,NEM,MXF
10132                                          ;       RMDS  - LBT
10133                                          ;       RMER1 - AOE
10134                                          ;NOTE:
10135                                          ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
10136                                          ;CYLINDER REGISTER IS WRITTEN
10137                                          ;NOTE:
10138                                          ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
10139                                          ;NOTE:
10140                                          ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
10141
10142
10143                                          ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10144  047354  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10145  047360  032737  001000  051526                  BIT     #AOE,S00$       ;ARE ERRORS ENABLED ??
10146  047366  001002                                  BNE     165$            ;YES !!
10147  047370  042716  100000                          BIC     #DLT,(SP)       ;RESET DLT ENABLE
10148  047374  013737  001336  001142  165$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10149  047402  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
10150  047406  001412                                  BEQ     170$            ;BRANCH IF DLT IS OK
10151  047410  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10152  047414  112776  000032  000000                  MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
10153  047422  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10154  047426  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10155  047430  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10156  047434                                 170$:
10157
10158                                          ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10159  047434  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10160  047440  032737  001000  051526                  BIT     #AOE,S00$       ;ARE ERRORS ENABLED ??
10161  047446  001002                                  BNE     175$            ;YES !!
10162  047450  042716  004000                          BIC     #NEM,(SP)       ;DISABLE NEM
10163  047454  013737  001336  001142  175$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10164  047462  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
10165  047466  001412                                  BEQ     180$            ;BRANCH IF NEM IS OK
10166  047470  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10167  047474  112776  000167  000000                  MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
10168  047502  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10169  047506  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10170  047510  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
10171  047514                                  180$:
10172
10173                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10174  047514  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10175  047520  032737  001000  051526                  BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
10176  047526  001002                                  BNE     185$            ;YES !!
10177  047530  042716  001000                          BIC     #MXF,(SP)       ;DISABLE MXF ERROR
10178  047534  013737  001336  001142  185$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10179  047542  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
10180  047546  001412                                  BEQ     190$            ;BRANCH IF MXF IS OK
10181  047550  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10182  047554  112776  000033  000000                  MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
10183  047562  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10184  047566  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10185  047570  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10186  047574                                  190$:
10187
10188                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
10189  047574  012746  177777                          MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
10190  047600  032737  001000  051526                  BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
10191  047606  001404                                  BEQ     191$            ;NO !!
10192  047610  032737  002000  001340                  BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
10193  047616  001002                                  BNE     195$            ;YES !!
10194  047620  042716  001000          191$:   BIC     #AOE,(SP)       ;DISABLE AOE
10195  047624  013737  001342  001142  195$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10196  047632  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
10197  047636  001412                                  BEQ     200$            ;BRANCH IF AOE IS OK
10198  047640  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10199  047644  112776  000020  000000                  MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
10200  047652  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10201  047656  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10202  047660  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10203  047664                                  200$:
10204
10205                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10206                                          ;HEADER ERRORS, I.E.,
10207                                          ;       RMER1 - HCRC,HCE,FER
10208                                          ;       RMER2 - BSE
10209
10210                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
10211  047664  032737  002000  001360                  BIT     #HCI,RMOFI      ;IS HCI SET ??
10212  047672  001403                                  BEQ     201$            ;NO !!
10213  047674  042737  000200  051526                  BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
10214  047702                                  201$:
10215
10216                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
10217  047702  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10218  047706  032737  000200  051526                  BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10219  047714  001002                                  BNE     205$            ;YES !!
10220  047716  042716  000400                          BIC     #HCRC,(SP)      ;DISABLE HCRC
10221  047722  013737  001342  001142  205$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10222  047730  042637  001142                          BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
10223  047734  001412                                  BEQ     210$            ;BRANCH IF HCRC IS OK
10224  047736  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10225  047742  112776  000035  000000                  MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
10226  047750  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
10227  047754  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10228  047756  162716  000C10                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10229  047762                          210$:
10230
10231                                  ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
10232  047762  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10233  047766  032737  000200  051526          BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
10234  047774  001002                          BNE     215$            ;YES !!
10235  047776  042716  000200                  BIC     #HCE,(SP)       ;DISABLE HCE
10236  050002  013737  001342  001142  215$:   MOV     RMER1I,$BODAT   ;GET RECEIVED STATUS
10237  050010  042637  001142                  BIC     (SP)+,$BODAT    ;CLEAR HCE IF ENABLED
10238  050014  001412                          BEQ     220$            ;BRANCH IF HCE IS OK
10239  050016  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10240  050022  112776  000036  000000          MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
10241  050030  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10242  050034  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10243  050040  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10244  050042                          220$:
10245
10246                                  ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
10247  050042  012746  177777                  MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
10248  050046  032737  000200  051526          BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10249  050054  001002                          BNE     225$            ;YES !!
10250  050056  042716  000020                  BIC     #FER,(SP)       ;DISABLE FER
10251  050062  013737  001342  001142  225$:   MOV     RMER1I,$BODAT   ;GET RECEIVED STATUS
10252  050070  042637  001142                  BIC     (SP)+,$BODAT    ;RESET FER IF ENABLED
10253  050074  001412                          BEQ     230$            ;BRANCH IF FER OK
10254  050076  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10255  050102  112776  000037  000000          MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
10256  050110  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10257  050114  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10258  050116  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10259  050122                          230$:
10260
10261                                  ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
10262  050122  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10263  050126  032737  000200  051526          BIT     #HCE,500$       ;ARE THEY ENABLED ??
10264  050134  001002                          BNE     235$            ;YES !!
10265  050136  042716  100000                  BIC     #BSE,(SP)       ;DISABLE BSE
10266  050142  013737  001370  001142  235$:   MOV     RMER2I,$BODAT   ;GET RECEIVED STATUS
10267  050150  042637  001142                  BIC     (SP)+,$BODAT    ;CLEAR BSE IF ENABLED
10268  050154  001412                          BEQ     240$            ;BRANCH IF BSE OK
10269  050156  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10270  050162  112776  000354  000000          MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
10271  050170  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10272  050174  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10273  050176  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10274  050202                          240$:
10275
10276                                  ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
10277                                  ;FIELD ERRORS, I.E.,
10278                                  ;       RMCS2 - MDPE
10279                                  ;       RMER1 - DCK,ECH
10280                                  ;NOTE:
10281                                  ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
10282                                  ;DCK IS SET.
```

```
10283
10284                                   ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
10285  050202  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10286  050206  032737  000100  051526         BIT     #ECH,5OO$       ;ARE DATA FIELD ERRORS ENABLED ??
10287  050214  001002                         BNE     245$            ;YES !!
10288  050216  042716  000400                 BIC     #MDPE,(SP)      ;DISABLE MDPE
10289  050222  013737  001336  001142 245$:   MOV     RMCS2I,$BODAT   ;GET RECEIVED STATUS
10290  050230  042637  001142                 BIC     (SP)+,$BODAT    ;CLEAR MDPE IF ENABLED
10291  050234  001412                         BEQ     250$            ;BRANCH IF MDPE OK
10292  050236  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10293  050242  112776  000027  000000         MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
10294  050250  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10295  050254  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10296  050256  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10297  050262                          250$:
10298
10299                                   ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
10300  050262  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10301  050266  032737  000100  051526         BIT     #ECH,5OO$       ;ARE THEY ENABLED ??
10302  050274  001002                         BNE     255$            ;YES !!
10303  050276  042716  100000                 BIC     #DCK,(SP)       ;DISABLE DCK
10304  050302  013737  001342  001142 255$:   MOV     RMER1I,$BODAT   ;GET RECEIVED STATUS
10305  050310  042637  001142                 BIC     (SP)+,$BODAT    ;CLEAR DCK IF ENABLED
10306  050314  001412                         BEQ     260$            ;BRANCH IF DCK IS OK
10307  050316  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10308  050322  112776  000030  000000         MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
10309  050330  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10310  050334  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10311  050336  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10312  050342                          260$:
10313
10314                                   ;REPORT ERROR IF ECH IS SET AND,
10315                                   ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
10316                                   ;       ECI IS SET, OR
10317                                   ;       DCK IS NOT SET.
10318  050342  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10319  050346  032737  000100  051526         BIT     #ECH,5OO$       ;ARE ERRORS ENABLED ??
10320  050354  001410                         BEQ     265$            ;NO !!
10321  050356  032737  004000  001360         BIT     #ECI,RMOFI      ;IS ECI SET ??
10322  050364  001004                         BNE     265$            ;YES !!
10323  050366  032737  100000  001342         BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
10324  050374  001002                         BNE     270$            ;YES !!
10325  050376  042716  000100          265$:  BIC     #ECH,(SP)       ;DISABLE ECH
10326  050402  013737  001342  001142 270$:   MOV     RMER1I,$BODAT   ;GET RECEIVED STATUS
10327  050410  042637  001142                 BIC     (SP)+,$BODAT    ;CLEAR ECH IF ENABLED
10328  050414  001412                         BEQ     275$            ;BRANCH IF ECH IS OK
10329  050416  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10330  050422  112776  000031  000000         MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
10331  050430  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10332  050434  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10333  050436  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10334  050442                          275$:
10335
10336                                   ;;*****************************************************************
10337                                   ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
10338
```

```
 10339  050442  022737  000030  051534          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 10340  050450  103402                          BLO     280$            ;YES!!
 10341  050452  000137  051500                  JMP     355$
 10342
```

```
10343                                   ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
10344                                   ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10345  050456  013737  001330  001142  280$:   MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
10346  050464  001421                           BEQ     285$            ;YES
10347  050466  032737  040000  001326           BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
10348  050474  001015                           BNE     285$            ;YES!!
10349  050476  062716  000004                   ADD     #4,(SP)
10350  050502  112776  000015  000000           MOVB    #15,@(SP)       ;ERROR NUMBER
10351  050510  005037  001140                   CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
10352  050514  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10353  050520  004736                           JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
10354  050522  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
10355  050526  000240                           NOP
10356                                   ;REPORT ERROR IF RMBA IS NOT CORRECT
10357  050530  013737  001330  001140  285$:   MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
10358  050536  163737  001400  001140           SUB     RMWCO,$GDDAT
10359  050544  006337  001140                   ASL     $GDDAT
10360  050550  063737  001402  001140           ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
10361  050556  032737  000010  001336           BIT     #BAI,RMCS2I     ;WAS BAI SET ??
10362  050564  001403                           BEQ     290$            ;NO !!
10363  050566  013737  001402  001140           MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
10364  050574  023737  001140  001332  290$:   CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
10365  050602  001416                           BEQ     295$            ;YES!!
10366  050604  013737  001332  001142           MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
10367  050612  062716  000004                   ADD     #4,(SP)
10368  050616  112776  000016  000000           MOVB    #16,@(SP)       ;ERROR NUMBER
10369  050624  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10370  050630  004736                           JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
10371  050632  162716  000010                   SUB     #10,(SP)                ;RESTORE (SP)
10372  050636  000240                           NOP
10373                                   ;COMPUTE NUMBER OF SECTORS TRANSFERRED
10374  050640  005046                   295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
10375  050642  013746  001330                   MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
10376  050646  163716  001400                   SUB     RMWCO,(SP)
10377  050652  012746  000400                   MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
10378  050656  032737  000002  001376           BIT     #BIT1,RMCS10    ;HEADER & DATA COMMAND ??
10379  050664  001402                           BEQ     300$            ;NO !!
10380  050666  012716  000402                   MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
10381  050672  005266  000004           300$:   INC     4(SP)           ;INCREMENT SECTOR COUNT
10382  050676  161666  000002                   SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
10383  050702  003373                           BGT     300$            ;CONTINUE IF NOT DONE
10384  050704  022626                           CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
10385                                   ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
10386  050706  013737  001404  051532           MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
10387  050714  013737  001404  051530           MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
10388  050722  013737  001432  051526           MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
10389  050730  042737  177740  051532           BIC     #^C37,510$
10390  050736  000337  051530                   SWAB    505$
10391  050742  042737  177770  051530           BIC     #^C7,505$
10392  050750  062637  051532                   ADD     (SP)+,510$
10393
10394  050754  023727  051532  000040  305$:   CMP     510$,#32.       ;SECTOR OVEFLOWED??
10395  050762  103420                           BLO     315$            ;NO!!
10396  050764  023727  051532  000240           CMP     510$,#(5*32.)   ;CYLINDERS WORTH??
10397  050772  103406                           BLO     310$            ;NO!!
10398  050774  005237  051526                   INC     500$            ;YES INCREMENT CYLINDER
```

```
10399  051000  162737  000240  051532          SUB     #(5*32.),510$   ;ADJUST SECTOR
10400  051006  000762                          BR      305$            ;TRY AGAIN
10401  051010  005237  051530          310$:   INC     505$            ;INCREMENT TRACK
10402  051014  162737  000040  051532          SUB     #32.,510$       ;ADJUST SECTOR
10403  051022  000754                          BR      305$            ;TRY AGAIN
10404
10405  051024  023727  051530  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
10406  051032  103406                          BLO     320$            ;NO!!
10407  051034  005237  051526                  INC     500$            ;INCREMENT CYLINDER
10408  051040  162737  000005  051530          SUB     #5,505$ ;ADJUST TRACK
10409  051046  000766                          BR      315$            ;TRY AGAIN
10410                                  ;REPORT ERROR IF "LBT" IS NOT CORRECT
10411  051050  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
10412  051054  022737  001467  051526          CMP     #823.,500$      ;SHOULD LBT BE SET??
10413  051062  101007                          BHI     325$            ;NO!!
10414  051064  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS IAE SET ??
10415  051072  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
10416  051074  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
10417  051102  013737  001340  001142  325$:   MOV     RMOS1,$BDDAT    ;BAD DATA FOR TYPEOUT
10418  051110  042737  175777  001142          BIC     #^CLBT,$BDDAT
10419  051116  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
10420  051124  001413                          BEQ     330$            ;YES!!
10421  051126  062716  000004                  ADD     #4,(SP)
10422  051132  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
10423  051140  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10424  051144  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
10425  051146  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10426  051152  000240                          NOP
10427                                  ;REPORT ERROR IF "AOE" IS INCORRECT
10428  051154  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
10429  051160  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
10430  051166  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
10431  051170  022737  001467  051526          CMP     #823.,500$      ;SHOULD AOE BE SET??
10432  051176  101025                          BHI     340$            ;NO!!
10433  051200  005737  051530                  TST     505$            ;MAYBE
10434  051204  001012                          BNE     335$            ;YES
10435  051206  005737  051532                  TST     510$
10436  051212  001007                          BNE     335$            ;YES !!
10437  051214  032737  000010  051534          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
10438  051222  001413                          BEQ     340$            ;NO !!
10439  051224  005737  001330                  TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
10440  051230  001410                          BEQ     340$            ;YES !!
10441  051232  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
10442  051240  005037  051530                  CLR     505$            ;CLEAR EXPECTED TRACK
10443  051244  012737  000001  051532          MOV     #1,510$ ;EXPECT SECTOR=1
10444  051252  013737  001342  001142  340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10445  051260  042737  176777  001142          BIC     #^CAOE,$BDDAT
10446  051266  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
10447  051274  001413                          BEQ     345$            ;YES!!
10448  051276  062716  000004                  ADD     #4,(SP)
10449  051302  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
10450  051310  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10451  051314  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
10452  051316  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10453  051322  000240                          NOP
10454                                  ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
10455  051324  032737  002000  001342  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
10456  051332  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
10457  051334  013737  051530  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
10458  051342  000337  001140                  SWAB    $GDDAT
10459  051346  113737  051532  001140          MOVB    510$,$GDDAT
10460  051354  013737  001334  001142          MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
10461  051362  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10462  051370  001413                          BEQ     350$            ;BRANCH IF EQUAL
10463  051372  062716  000004                  ADD     #4,(SP)
10464  051376  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
10465  051404  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10466  051410  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
10467  051412  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10468  051416  000240                          NOP
10469                                  ;REPORT ERROR IF RMDC IS INCORRECT
10470  051420  013737  051526  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
10471  051426  042737  176000  001140          BIC     #^C1777,$GDDAT
10472  051434  013737  001362  001142          MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
10473  051442  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
10474  051450  001413                          BEQ     355$            ;BRANCH IF EQUAL
10475  051452  062716  000004                  ADD     #4,(SP)
10476  051456  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
10477  051464  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10478  051470  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
10479  051472  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10480  051476  000240                          NOP
```

```
10481  051500  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10482  051504  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
10483  051510  001403                          BEQ     360$
10484  051512  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
10485  051516  000402                          BR      365$
10486  051520  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10487  051524  000207                  365$:   RTS     PC
10488
10489  051526  000000                  500$:   .WORD   0               ;CYLINDER
10490  051530  000000                  505$:   .WORD   0               ;TRACK
10491  051532  000000                  510$:   .WORD   0               ;SECTOR
10492  051534  000000                  515$:   .WORD   0               ;FUNCTION CODE
10493
10494
```

```
 10495                                  .SBTTL  DEVICE SELECT SUBROUTINE
 10496
 10497                                  ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
 10498                                  ; TEST QUEUE.
 10499
 10500                                  ;CALL:
 10501                                  ;(1)    JSR     PC,DEVSEL
 10502                                  ;(2)    BR      ??              RETURN IF NO ERROR
 10503                                  ;(3)    NOP                     RETURN IF ERROR
 10504                                  ;(4)    ERROR                   ERROR DEFINED BY SUBROUTINE
 10505
 10506  051536                          DEVSEL:
 10507
 10508                                  ;CLEAR USER'S ERROR CALL
 10509  051536  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 10510  051542  105076  000000                 CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
 10511  051546  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK
 10512                                  ;SAVE USER'S INFORMATION AND SETUP REGISTERS
 10513  051552  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 10514  051556  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
 10515  051562  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 10516  051564  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 10517  051566  012737  051706  000004         MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
 10518  051574  012737  000300  000006         MOV     #PR6,ERRVEC+2
 10519  051602  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
 10520  051606  013701  001446                 MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER
 10521
 10522                                  ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
 10523  051612  111160  000010                 MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
 10524  051616  016037  000000  001176         MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
 10525  051624  016037  000010  001174         MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
 10526  051632  032737  010000  001174         BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
 10527  051640  001407                         BEQ     10$             ;NO!!
 10528  051642  062766  000004  000010         ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
 10529  051650  112776  000111  000010         MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
 10530  051656  000422                         BR      30$             ;
 10531  051660  032737  004000  001176  10$:    BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
 10532  051666  001021                         BNE     35$             ;YES!!
 10533  051670  062766  000004  000010         ADD     #4,10(SP)
 10534  051676  112776  000112  000010         MOVB    #112,@10(SP)
 10535  051704  000407                         BR      30$
 10536
 10537  051706                          20$:
 10538
 10539                                  ;HANDLE BUS TIMEOUT
 10540  051706  022626                         CMP     (SP)+,(SP)+     ;ADJUST SP
 10541  051710  062766  000004  000010         ADD     #4,10(SP)
 10542  051716  112776  000113  000010         MOVB    #113,@10(SP)
 10543  051724  162766  000002  000010  30$:    SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR
 10544
 10545  051732                          35$:
 10546
 10547
 10548                                  ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
 10549  051732  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
 10550  051734  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 10551  051736  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 10552  051742  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 10553  051746  000207                        RTS     PC              ;EXIT
```

```
10554                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
10555                                   ;
10556                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
10557                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
10558
10559                                   ;
10560                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
10561                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
10562                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
10563                                   ;
10564                                   ;CALL:
10565                                   ;(1)    JSR     PC,SEKSTS
10566                                   ;       BR      ???             RETURN HERE IF NO ERROR
10567                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10568                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10569                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10570                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10571
10572  051750                           SEKSTS:
10573
10574                                   ;CLEAR USER'S ERROR  CALL
10575  051750  000240                          NOP
10576  051752  062716  000004                 ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10577  051756  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
10578  051762  162716  000004                 SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10579  051766  005037  053206                 CLR     300$            ;CLEAR ERROR FLAGS
10580
10581                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
10582                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
10583  051772  032737  000010  001342           BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
10584  052000  001424                          BEQ     1$              ;NO!!
10585  052002  032737  000010  001370         BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
10586  052010  001020                          BNE     1$              ;NOT SURE!!
10587
10588                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
10589  052012  005037  001140                 CLR     $GDDAT          ;EXPECTED STATUS
10590  052016  013737  001342  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10591  052024  062716  000004                 ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
10592  052030  112776  000050  000000         MOVB    #50,@(SP)       ;ERROR #50
10593  052036  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10594  052042  004736                          JSR     PC,@(SP)+
10595  052044  162716  000010                 SUB     #10,(SP)        ;RESTORE STACK
10596  052050  000437                          BR      3$              ;IAE SHOULD  BE ZERO
10597
10598
10599                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
10600                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
10601  052052  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
10602  052060  052737  040000  053206         BIS     #SKI,300$       ;SET SKI ERROR FLAG
10603  052066  023727  001432  001466         CMP     RMDC0,#822.     ;CYLINDER > 822??
10604  052074  101025                          BHI     3$              ;YES!!
10605  052076  042737  040000  053206         BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
10606  052104  123727  001404  000037         CMPB    RMDA0,#31.      ;SECTOR > 31??
10607  052112  101016                          BHI     3$              ;YES!!
10608  052114  123727  001404  000035         CMPB    RMDA0,#29.      ;SECTOR > 29 ??
10609  052122  101404                          BLOS    2$              ;NO!!
```

```
10610  052124  032737  010000  001430          BIT     #FMT16,RMOF0    ;30 SECTOR FORMAT??
10611  052132  001406                          BEQ     3$              ;YES!!
10612  052134  123727  001405  000004  2$:     CMPB    RMDA0+1,#4.     ;TRACK  >4??
10613  052142  101002                          BHI     3$              ;YES!!
10614  052144  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
10615
10616                                          ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
10617  052150  013737  001342  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
10618  052156  042737  175777  001142          BIC     #↑CIAE,$BDDAT
10619  052164  023737  001140  001142          CMP     $GDDAT,$BDDAT
10620  052172  001004                          BNE     35$             ;IAE IN ERROR
10621  052174  042737  040000  053206          BIC     #SKI,300$       ;CLEAR SKI FLAG
10622  052202  000413                          BR      5$              ;GO CHECK NEXT ERROR
10623  052204                                  35$:
10624                                          ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
10625  052204  062716  000004                  ADD     #4,(SP)
10626  052210  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
10627  052216  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10628  052222  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
10629  052224  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10630  052230  000240                          NOP
10631  052232                                  5$:
10632
10633                                          ;REPORT ANY IVC ERROR AS
10634                                          ;       IVC ERROR WITH VOLUME VALID ZERO
10635                                          ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
10636  052232  032737  010000  001370          BIT     #IVC,RMER2I     ;IVC ERROR??
10637  052240  001427                          BEQ     52$             ;NO!!
10638  052242  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
10639  052246  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10640  052254  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10641  052260  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
10642  052266  032737  000100  001340          BIT     #VV,RMDSI
10643  052274  001403                          BEQ     51$
10644  052276  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
10645  052304  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10646  052310  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10647  052312  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10648  052316  000240                          NOP
10649
10650  052320  013737  001370  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10651  052326  042737  137777  001142          BIC     #↑CSKI,$BDDAT   ;CLEAR DONT CARES
10652  052334  013737  053206  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
10653  052342  042737  137777  001140          BIC     #↑CSKI,$GDDAT   ;CLEAR DONT CARES
10654  052350  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
10655
10656                                          ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
10657  052352  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
10658  052360  001032                          BNE     54$             ;YES !!
10659  052362  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10660  052366  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
10661  052374  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10662  052400  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10663  052402  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10664  052406  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
10665  052410                                  53$:
```

```
10666
10667                                   ;REPORT ERROR IF SKI IS SET
10668  052410  032737  040000  001142             BIT     #SKI,$BDDAT     ;IS SKI SET ??
10669  052416  001413                           BEQ     54$             ;NO - SKI IS OK
10670  052420  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10671  052424  112776  000054  000000           MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
10672  052432  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10673  052436  004736                           JSR     PC,@(SP)+       ;REPORT SEEK ERROR
10674  052440  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
10675  052444  000240                           NOP
10676
10677                                   ;REPORT ANY DEVICE CHECK
10678  052446  032737  000200  001370 54$:      BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
10679  052454  001420                           BEQ     6$              ;NO!!
10680  052456  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
10681  052462  013737  001370  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10682  052470  062716  000004                   ADD     #4,(SP)
10683  052474  112776  000055  000000           MOVB    #55,@(SP)       ;ERROR  #55
10684  052502  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10685  052506  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10686  052510  162716  000010                   SUB     #10,(SP)        ;RESTORE SP
10687  052514  000240                           NOP
10688
10689                                   ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
10690                                   ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
10691  052516  032737  020000  001342 6$:       BIT     #OPI,RMER1I     ;"OPI" ERROR??
10692  052524  001427                           BEQ     8$              ;NO!!
10693  052526  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
10694  052532  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10695  052540  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10696  052544  112776  000052  000000           MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
10697  052552  032737  010000  001340           BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
10698  052560  001403                           BEQ     7$              ;NO!!
10699  052562  112776  000053  000000           MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
10700  052570  162716  000002           7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10701  052574  004736                           JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
10702  052576  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
10703  052602  000240                           NOP
10704
10705                                   ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
10706  052604  013746  001340           8$:     MOV     RMDSI,-(SP)
10707  052610  042716  047677                   BIC     #^C<ATA!PIP!MOL!VV>,(SP)
10708  052614  022726  110100                   CMP     #ATA!MOL!VV,(SP)+
10709  052620  001002                           BNE     9$              ;ERROR IN RMDS
10710  052622  000137  053156                   JMP     14$             ;RMDS IS OK
10711
10712                                   ;REPORT ERROR IF MOL = 0 AND OPI = 0
10713  052626  032737  010000  001340 9$:       BIT     #MOL,RMDSI      ;IS MOL RESET??
10714  052634  001030                           BNE     10$             ;NO - MOL IS SET
10715  052636  032737  020000  001342           BIT     #OPI,RMER1I     ;WAS OPI SET
10716  052644  001024                           BNE     10$             ;YES - DONT REPORT ERROR
10717  052646  013737  001340  001140           MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10718  052654  052737  010000  001140           BIS     #MOL,$GDDAT
10719  052662  013737  001340  001142           MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10720  052670  062716  000004                   ADD     #4,(SP)
10721  052674  112776  000062  000000           MOVB    #62,@(SP)
```

```
10722  052702  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10723  052706  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10724  052710  162716  000010          SUB     #10,(SP)
10725  052714  000240                  NOP
10726
10727                          ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
10728  052716  032737  020000  001340  10$:    BIT     #PIP,RMDSI      ;IS "PIP" STILL SET??
10729  052724  001430                  BEQ     11$             ;NO!!
10730  052726  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI"SET??
10731  052734  001024                  BNE     11$             ;YES-DONT REPORT PIP
10732  052736  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10733  052744  042737  020000  001142          BIC     #PIP,$BDDAT
10734  052752  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10735  052760  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10736  052764  112776  000056  000000          MOVB    #56,@(SP)       ;LOAD ERROR NUMBER
10737  052772  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10738  052776  004736                  JSR     PC,@(SP)+       ;REPORT "PIP" SET AFTER SEEK
10739  053000  162716  000010          SUB     #10,(SP)                ;RESTORE (SP)
10740  053004  000240                  NOP
10741
10742                          ;REPORT AN ERROR IF "ATA" IS NOT SET
10743  053006  032737  100000  001340  11$:    BIT     #ATA,RMDSI      ;WAS "ATA" SET ??
10744  053014  001024                  BNE     13$             ;YES!!
10745  053016  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10746  053024  052737  110600  001140          BIS     #ATA!MOL!DPR!DRY,$GDDAT
10747  053032  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10748  053040  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10749  053044  112776  000057  000000          MOVB    #57,@(SP)       ;LOAD ERROR NUMBER
10750  053052  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10751  053056  004736                  JSR     PC,@(SP)+       ;REPORT ATTENTION NOT SET DURING
10752                                          ;SEEK TEST
10753  053060  162716  000010          SUB     #10,(SP)                ;RESTORE (SP)
10754  053064  000240                  NOP
10755
10756                          ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
10757  053066  032737  000100  001340  13$:    BIT     #VV,RMDSI       ;IS VV = 0 ??
10758  053074  001030                  BNE     14$             ;NO!!
10759  053076  032737  010000  001370          BIT     #IVC,RMER2I     ;IS IVC ALSO 0 ??
10760  053104  001024                  BNE     14$             ;NO - IVC IS SET
10761  053106  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10762  053114  052737  000100  001140          BIS     #VV,$GDDAT
10763  053122  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10764  053130  062716  000004                  ADD     #4,(SP)
10765  053134  112776  000064  000000          MOVB    #64,@(SP)       ;ERROR #64
10766  053142  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10767  053146  004736                  JSR     PC,@(SP)+
10768  053150  162716  000010          SUB     #10,(SP)
10769  053154  000240                  NOP
10770  053156                  14$:
10771
10772                          ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10773  053156  000240                  NOP
10774  053160  062716  000004          ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10775  053164  105776  000000          TSTB    @(SP)           ;WAS ERROR CALLED??
10776  053170  001403                  BEQ     15$             ;NO!!
10777  053172  062716  000004          ADD     #4,(SP)         ;MOVE TO ERROR RETURN
```

```
10778  053176  000402                    BR      16$
10779
10780  053200  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10781  053204  000207              16$:    RTS     PC              ;RETURN
10782
10783  053206  000000              300$:   .WORD   0               ;ERROR FLAGS
10784
```

```
10785                                           .SBTTL  CONTROLLER CLEAR SUBROUTINE
10786
10787                                           ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
10788                                           ;AND DRIVES, THEN SELECTS THE DRIVE.
10789                                           ;
10790                                           ;CALL:  JSR     PC,CNTCLR
10791                                           ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10792                                           ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10793                                           ;       ERROR                   SUB DEFINES ERROR NUMBER
10794                                           ;       ???
10795
10796   053210                                  CNTCLR:
10797   053210  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10798   053212  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10799   053214  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10800   053220  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10801   053224  012737  053264  000004                  MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
10802   053232  012737  000300  000006                  MOV     #PR6,ERRVEC+2
10803   053240  013700  001276                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
10804   053244  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
10805   053252  013701  001446                          MOV     TSTQUE,R1
10806   053256  111160  000010                          MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
10807   053262  000412                                  BR      20$
10808   053264  022626                          10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
10809   053266  062766  000004  000010                  ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
10810   053274  112776  000007  000010                  MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
10811   053302  162766  000002  000010                  SUB     #2,10(SP)
10812   053310                                  20$:
10813   053310  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10814   053314  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10815   053320  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
10816   053322  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
10817   053324  000207                                  RTS     PC
10818
```

```
10819                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
10820
10821                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
10822                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
10823                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
10824                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
10825
10826                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
10827                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10828                                   ;
10829                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
10830                                   ;
10831
10832                                   ;CALL:
10833                                   ;(1)    JSR     PC,CLRSTS
10834                                   ;       BR      ???             RETURN HERE IF NO ERROR
10835                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10836                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10837                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10838                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10839
10840  053326                           CLRSTS:
10841
10842                                   ;CLEAR USER'S ERROR CALL
10843  053326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR
10844  053332  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
10845  053336  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
10846                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10847  053342  013737  001326  001142  4$:     MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
10848  053350  042737  100000  001142          BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
10849  053356  012737  004200  001140          MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
10850  053364  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
10851  053372  001413                          BEQ     5$              ;BRANCH IF EQUAL
10852  053374  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10853  053400  112776  000126  000000          MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
10854  053406  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10855  053412  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10856  053414  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10857  053420  000240                          NOP
10858                                   ;REPORT ERROR IF RMBA NOT RESET
10859  053422  005037  001140          5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
10860  053426  013737  001332  001142          MOV     RMBAI,$BDDAT
10861  053434  001413                          BEQ     7$              ;BRANCH IF ZERO
10862  053436  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10863  053442  112776  000127  000000          MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
10864  053450  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10865  053454  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10866  053456  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10867  053462  000240                          NOP
10868                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
10869  053464  013737  001336  001142  7$:     MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
10870  053472  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10871  053474  005046                          CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
10872  053476  013701  001446                  MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
10873  053502  111116                          MOVB    (R1),(SP)
10874  053504  052716  000100                  BIS     #IR,(SP)
```

```
10875  053510  012637  001140                  MOV     (SP)+,$GDDAT
10876  053514  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10877  053516  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10878  053524  001413                          BEQ     9$              ;BRANCH IF EQUAL
10879  053526  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10880  053532  112776  000130  000000          MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
10881  053540  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10882  053544  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10883  053546  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10884  053552  000240                          NOP
10885                                  ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
10886  053554  005037  001140          9$:     CLR     $GDDAT          ;VERIFY RMER1
10887  053560  013737  001342  001142          MOV     RMER1I,$BDDAT
10888  053566  042737  040000  001142          BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
10889  053574  001413                          BEQ     13$             ;BRANCH IF ZERO
10890  053576  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10891  053602  112776  000131  000000          MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
10892  053610  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10893  053614  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10894  053616  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10895  053622  000240                          NOP
10896                                  ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
10897  053624  013737  001352  001142  13$:     MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
10898  053632  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
10899  053640  012737  000010  001140          MOV     #WWD,$GDDAT     ;EXPECT WRITE DATA BIT
10900  053646  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10901  053654  001413                          BEQ     17$             ;BRANCH IF 0
10902  053656  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10903  053662  112776  000133  000000          MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
10904  053670  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10905  053674  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10906  053676  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10907  053702  000240                          NOP
10908                                  ;REPORT AN ERROR IF RMEC2 IS NOT RESET
10909  053704  005037  001140          17$:    CLR     $GDDAT          ;EXPECT ZEROS
10910  053710  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
10911  053716  001413                          BEQ     19$
10912  053720  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10913  053724  112776  000135  000000          MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
10914  053732  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10915  053736  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10916  053740  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10917  053744  000240                          NOP
10918                                  ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
10919  053746  013737  001366  001142  19$:     MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
10920  053754  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
10921  053762  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10922  053770  023737  001140  001142          CMP     $GDDAT,$BDDAT
10923  053776  001413                          BEQ     21$
10924  054000  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10925  054004  112776  000136  000000          MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
10926  054012  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10927  054016  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10928  054020  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10929  054024  000240                          NOP
10930                                  ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
10931  054026  005037  001140            21$:    CLR     $GDDAT              ;EXPECT ALL ZEROS
10932  054032  013737  001370  001142            MOV     RMER2I,$BDDAT       ;VERIFY RMER2
10933  054040  042737  040200  001142            BIC     #SKI!DVC,$BDDAT     ;IGNORE DEVICE ERRORS
10934  054046  001413                            BEQ     215$                ;BRANCH IF OTHER BITS 0
10935  054050  062716  000004                    ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10936  054054  112776  000174  000000            MOVB    #174,2(SP)          ;WITE ERROR NUMBER IN CALL
10937  054062  162716  000002                    SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10938  054066  004736                            JSR     PC,2(SP)+           ;REPORT ERROR VIA USER
10939  054070  162716  000010                    SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10940  054074  000240                            NOP
10941                                    ;REPORT ERROR IF RMDS NOT INITIALIZED
10942  054076  013737  001340  001142    215$:   MOV     RMDSI,$BDDAT        ;TEST DRIVE STATUS REGISTER
10943  054104  042737  177177  001142            BIC     #^C<DRY!DPR>,$BDDAT
10944  054112  012737  000600  001140            MOV     #DPR!DRY,$GDDAT     ;EXPECTED DRIVE STATUS
10945  054120  023737  001140  001142            CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
10946  054126  001413                            BEQ     22$                 ;BRANCH IF EQUAL
10947  054130  062716  000004                    ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10948  054134  112776  000134  000000            MOVB    #134,2(SP)          ;WRITE ERROR NUMBER
10949  054142  162716  000002                    SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10950  054146  004736                            JSR     PC,2(SP)+           ;REPORT ERROR TO USER
10951  054150  162716  000010                    SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10952  054154  000240                            NOP
10953  054156  062716  000004            22$:    ADD     #4,(SP)             ;MOVE SP TO ERROR CALL
10954  054162  105776  000000                    TSTB    2(SP)               ;WAS AN ERROE DETECTED??
10955  054166  001403                            BEQ     23$                 ;NO!!
10956  054170  062716  000004                    ADD     #4,(SP)             ;YES - MOVE TO ERROR RETURN
10957  054174  000402                            BR      24$
10958  054176  162716  000004            23$:    SUB     #4,(SP)             ;MOVE SP TO NO ERROR RETURN
10959  054202  000240                    24$:    NOP
10960  054204  000207                            RTS     PC
```

```
10961                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
10962
10963                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
10964                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
10965                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
10966
10967                                   ;CALL:
10968                                   ;(1)    JSR     PC,ACKSTS
10969                                   ;       BR      ???             RETURN HERE IF NO ERROR
10970                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10971                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10972                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10973                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10974
10975   054206                          ACKSTS:
10976
10977                                   ;CLEAR USER'S ERROR CALL
10978   054206  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10979   054212  105076  000000                CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
10980   054216  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK
10981
10982                                   ;REPORT AN ERROR  IF "VV" IS 0
10983   054222  032737  000100  001340        BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
10984   054230  001024                        BNE     1$              ;YES!!
10985   054232  013737  001340  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10986   054240  052737  000100  001140        BIS     #VV,$GDDAT
10987   054246  013737  001340  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10988   054254  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10989   054260  112776  000155  000000        MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
10990   054266  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10991   054272  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR
10992   054274  162716  000010                SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10993   054300  000240                        NOP
10994   054302                          1$:
10995
10996                                   ;REPORT AN ERROR IF "MOL" IS 0
10997   054302  032737  010000  001340        BIT     #MOL,RMDSI      ;IS MOL SET??
10998   054310  001024                        BNE     2$              ;YES!!
10999   054312  013737  001340  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11000   054320  052737  010000  001140        BIS     #MOL,$GDDAT
11001   054326  013737  001340  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11002   054334  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11003   054340  112776  000041  000000        MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11004   054346  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11005   054352  004736                        JSR     PC,@(SP)+       ;REPORT TH ERROR
11006   054354  162716  000010                SUB     #10,(SP)        ;MOVE SP TO BRANCH
11007   054360  000240                        NOP
11008   054362                          2$:
11009
11010                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
11011   054362  032737  060007  001342        BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
11012   054370  001570                        BEQ     7$
11013
11014                                   ;REPORT AN ERROR IF "UNS" IS SET
11015   054372  032737  040000  001342        BIT     #UNS,RMER1I     ;WAS UNS SET??
11016   054400  001424                        BEQ     3$              ;NO!!
```

```
11017  054402  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11018  054410  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11019  054416  042737  040000  001140          BIC     #UNS,$GDDAT
11020  054424  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11021  054430  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11022  054436  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11023  054442  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11024  054444  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11025  054450  000240                          NOP
11026  054452                          3$:
11027
11028                                  ;REPORT ANY OPI ERROR
11029  054452  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET ??
11030  054460  001424                          BEQ     4$              ;NO!!
11031  054462  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11032  054470  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11033  054476  042737  020000  001140          BIC     #OPI,$GDDAT
11034  054504  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11035  054510  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11036  054516  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11037  054522  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11038  054524  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11039  054530  000240                          NOP
11040  054532                          4$:
11041
11042                                  ;REPORT ANY RMR ERROR
11043  054532  032737  000004  001342          BIT     #RMR,RMER1I     ;WAS RMR SET??
11044  054540  001424                          BEQ     5$              ;NO!!
11045  054542  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11046  054550  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11047  054556  042737  000004  001140          BIC     #RMR,$GDDAT
11048  054564  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11049  054570  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11050  054576  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11051  054602  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11052  054604  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11053  054610  000240                          NOP
11054  054612                          5$:
11055
11056                                  ;REPORT ANY     ILR ERROR
11057  054612  032737  000002  001342          BIT     #ILR,RMER1I     ;WAS ILR SET??
11058  054620  001424                          BEQ     6$              ;NO!!
11059  054622  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11060  054630  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11061  054636  042737  000002  001140          BIC     #ILR,$GDDAT
11062  054644  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11063  054650  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11064  054656  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11065  054662  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11066  054664  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11067  054670  000240                          NOP
11068  054672                          6$:
11069
11070                                  ;REPORT ANY     ILF ERROR
11071  054672  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS ILF SET??
11072  054700  001424                          BEQ     7$              ;NO!!
```

```
11073  054702  013737  001342  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
11074  054710  013737  001342  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
11075  054716  042737  000001  001140          BIC     #ILF,$GDDAT
11076  054724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11077  054730  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11078  054736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11079  054742  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11080  054744  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11081  054750  000240                          NOP
11082  054752                          7$:
11083
11084                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
11085  054752  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11086  054756  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11087  054762  001403                          BEQ     8$              ;NO!!
11088  054764  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11089  054770  000402                          BR      9$
11090  054772  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11091  054776  000240                  9$:     NOP
11092  055000  000207                          RTS     PC
11093
```

```
 11094                                  .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
 11095
 11096                                  ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
 11097                                  ;USING THE STATUS STORED IN THE GET BUFFER.
 11098
 11099                                  ;CALL:
 11100                                  ;
 11101                                  ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
 11102                                  ;       BR      ???             ;RETURN HERE IF NO ERROR
 11103                                  ;       NOP                     ;RETURN HERE TO REPORT AN ERROR
 11104                                  ;       ERROR                   ;ERROR NUMBER DEFINED BY SUB
 11105                                  ;       JSR     PC,@(SP)+       ;GO BACK TO SUB FOR MORE ERROR CHECKS
 11106                                  ;       ???                     ;RETURN HERE IF NO MORE ERRORS
 11107
 11108  055002                          RCLSTS:
 11109
 11110                                  ;CLEAR USER'S ERROR NUMBER
 11111  055002  062716  000004                  ADD     #4,(SP)
 11112  055006  105076  000000                  CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
 11113  055012  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
 11114
 11115
 11116                                  ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
 11117  055016  032737  022011  001342          BIT     #OPI!PAR!ILF!IAE,RMER1I
 11118  055024  001553                          BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
 11119
 11120                                  ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
 11121                                  ;"PAR" = 1 AND "DPE" = 0
 11122  055026  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS "PAR" SET??
 11123  055034  001430                          BEQ     1$              ;NO!!
 11124  055036  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS "DPE" SET??
 11125  055044  001024                          BNE     1$              ;YES - NOT A REGISTER ERROR
 11126  055046  013737  001342  001140          MOV RMER1I,$GDDAT       ;EXPECTED STATUS
 11127  055054  042737  000010  001140          BIC     #PAR,$GDDAT
 11128  055062  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 11129  055070  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 11130  055074  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
 11131  055102  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11132  055106  004736                          JSR     PC,@(SP)+       ;GO REPORT ERROR
 11133  055110  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
 11134  055114  000240                          NOP
 11135  055116                          1$:
 11136
 11137                                  ;REPORT ANY "ILF" ERROR
 11138  055116  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS "ILF" SET??
 11139  055124  001424                          BEQ     2$              ;NO!!
 11140  055126  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 11141  055134  042737  000001  001140          BIC     #ILF,$GDDAT
 11142  055142  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 11143  055150  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 11144  055154  112776  000071  000000          MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
 11145  055162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 11146  055166  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 11147  055170  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
 11148  055174  000240                          NOP
 11149  055176                          2$:
```

```
11150
11151                                   ;REPORT ANY "OPI" ERROR AS
11152                                   ;       . OPI DUE TO "MOL" = 0
11153                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
11154  055176  032737  020000  001342           BIT     #OPI,RMERII     ;WAS OPI SET??
11155  055204  001433                           BEQ     31$             ;NO!!
11156  055206  013737  001342  001140           MOV     RMERII,$GDDAT   ;EXPECTED STATUS
11157  055214  042737  020000  001140           BIC     #OPI,$GDDAT
11158  055222  013737  001342  001142           MOV     RMERII,$BDDAT   ;RECEIVED STATUS
11159  055230  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11160  055234  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11161  055242  032737  010000  001340           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
11162  055250  001403                           BEQ     3$              ;YES!!
11163  055252  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
11164  055260  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11165  055264  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11166  055266  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11167  055272  000240                           NOP
11168  055274                           31$:
11169
11170                                   ;REPORT AN ERROR IF "IAE" IS SET
11171  055274  032737  002000  001342           BIT     #IAE,RMERII     ;IS "IAE" SET??
11172  055302  001424                           BEQ     4$              ;NO!!
11173  055304  013737  001342  001140           MOV     RMERII,$GDDAT   ;EXPECTED STATUS
11174  055312  042737  002000  001140           BIC     #IAE,$GDDAT
11175  055320  013737  001342  001142           MOV     RMERII,$BDDAT   ;RECEIVED STATUS
11176  055326  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11177  055332  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11178  055340  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11179  055344  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
11180  055346  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
11181  055352  000240                           NOP
11182  055354                           4$:
11183
11184                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
11185  055354  032737  050200  001370           BIT     #SKI!IVC!DVC,RMER2I
11186  055362  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
11187
11188
11189                                   ;REPORT ANY "IVC" ERROR AS
11190                                   ;       .IVC WITH VV = 0
11191                                   ;       .ERRONEOUS IVC ERROR
11192  055364  032737  010000  001370           BIT     #IVC,RMER2I     ;WAS IVC SET??
11193  055372  001433                           BEQ     6$              ;NO!!
11194  055374  013737  001370  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11195  055402  042737  010000  001140           BIC     #IVC,$GDDAT
11196  055410  013737  001370  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11197  055416  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11198  055422  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
11199  055430  032737  000100  001340           BIT     #VV,RMDSI       ;WAS VV = 0??
11200  055436  001403                           BEQ     5$              ;YES!!
11201  055440  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
11202  055446  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11203  055452  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11204  055454  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11205  055460  000240                           NOP
```

```
11206  055462                                  6$:
11207
11208                                          ;REPORT ANY "SKI" ERROR
11209  055462  032737  040000  001370                BIT     #SKI,RMER2I     ;WAS SKI SET??
11210  055470  001424                              BEQ     7$              ;NO!!
11211  055472  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11212  055500  042737  040000  001140              BIC     #SKI,$GDDAT
11213  055506  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11214  055514  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11215  055520  112776  000076  000000              MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
11216  055526  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11217  055532  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11218  055534  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO BRANCH
11219  055540  000240                              NOP
11220  055542                                  7$:
11221
11222                                          ;REPORT ANY "DVC" ERROR
11223  055542  032737  000200  001370                BIT     #DVC,RMER2I     ;WAS "DVC" SET??
11224  055550  001424                              BEQ     8$              ;NO!!
11225  055552  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11226  055560  042737  000200  001140              BIC     #DVC,$GDDAT
11227  055566  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11228  055574  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11229  055600  112776  000077  000000              MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
11230  055606  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11231  055612  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11232  055614  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11233  055620  000240                              NOP
11234  055622                                  8$:
11235
11236                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
11237  055622  013746  001340                      MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
11238  055626  042716  047676                      BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
11239  055632  022726  110100                      CMP     #ATA!MOL!VV,(SP)+
11240  055636  001002                              BNE     85$
11241  055640  000137  056254                      JMP     13$
11242  055644                                  85$:
11243
11244                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
11245                                          ;LINE AFTER RECALIBRATE WAS INITIATED
11246  055644  032737  010000  001340                BIT     #MOL,RMDSI      ;DID MOL DROP??
11247  055652  001030                              BNE     9$              ;NO!!
11248  055654  032737  020000  001342              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11249  055662  001024                              BNE     9$              ;YES - DON'T REPORT MOL=0
11250  055664  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11251  055672  052737  010000  001140              BIS     #MOL,$GDDAT
11252  055700  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11253  055706  062716  000004                      ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11254  055712  112776  000100  000000              MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
11255  055720  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11256  055724  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11257  055726  162716  000010                      SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11258  055732  000240                              NOP
11259  055734                                  9$:
11260
11261                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
11262  055734  032737  000100  001340          BIT     #VV,RMDSI       ;DID "VV" DROP??
11263  055742  001030                          BNE     10$             ;NO!!
11264  055744  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
11265  055752  001024                          BNE     10$             ;YES - DONT REPORT VV=0
11266  055754  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11267  055762  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11268  055770  052737  000100  001140          BIS     #VV,$GDDAT
11269  055776  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11270  056002  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
11271  056010  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11272  056014  004736                          JSR     PC,@(SP)+
11273  056016  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11274  056022  000240                          NOP
11275  056024                          10$:
11276
11277                                  ;REPORT AN ERROR IF ATA IS NOT SET
11278  056024  032737  100000  001340          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
11279  056032  001024                          BNE     11$             ;YES!!
11280  056034  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11281  056042  052737  100000  001140          BIS     #ATA,$GDDAT
11282  056050  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11283  056056  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11284  056062  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
11285  056070  162716  000002                  SUB     #2,(SP)
11286  056074  004736                          JSR     PC,@(SP)+
11287  056076  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11288  056102  000240                          NOP
11289
11290  056104                          11$:
11291
11292                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
11293                                  ;ALWAYS CLEAR OFFSET MODE
11294  056104  032737  000001  001340          BIT     #OM,RMDSI       ;WAS "OM" RESET??
11295  056112  001424                          BEQ     12$             ;YES!!
11296  056114  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11297  056122  042737  000001  001140          BIC     #OM,$GDDAT
11298  056130  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11299  056136  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11300  056142  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
11301  056150  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11302  056154  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11303  056156  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11304  056162  000240                          NOP
11305  056164                          12$:
11306
11307                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
11308                                  ;CYLINDER
11309  056164  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11310  056172  001430                          BEQ     13$             ;NO!!
11311  056174  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11312  056202  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
11313  056204  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11314  056212  042737  020000  001140          BIC     #PIP,$GDDAT
11315  056220  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11316  056226  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11317  056232  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
11318  056240  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11319  056244  004736                  JSR     PC,@(SP)+
11320  056246  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11321  056252  000240                  NOP
11322  056254                  13$:
11323
11324                          ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
11325  056254  032737  040006  001342  BIT     #ILR!RMR!UNS,RMER1I
11326  056262  001514                  BEQ     16$
11327
11328                          ;REPORT AN ERROR IF "ILR" IS SET
11329  056264  032737  000002  001342  BIT     #ILR,RMER1I     ;WAS ILR SET DURING RECALIBRATE??
11330  056272  001424                  BEQ     14$             ;NO!!
11331  056274  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11332  056302  042737  000002  001140  BIC     #ILR,$GDDAT
11333  056310  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11334  056316  062716  000004          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11335  056322  112776  000105  000000  MOVB    #105,@(SP)      ;WRITE ERROR NUMBER IN CALL
11336  056330  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11337  056334  004736                  JSR     PC,@(SP)+
11338  056336  162716  000010          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11339  056342  000240                  NOP
11340  056344                  14$:
11341
11342                          ;REPORT AN ERROR IF "RMR" IS SET
11343  056344  032737  000004  001342  BIT     #RMR,RMER1I     ;WAS RMR SET??
11344  056352  001424                  BEQ     15$             ;NO!!
11345  056354  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11346  056362  042737  000004  001140  BIC     #RMR,$GDDAT
11347  056370  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11348  056376  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11349  056402  112776  000106  000000  MOVB    #106,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11350  056410  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11351  056414  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11352  056416  162716  000010          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11353  056422  000240                  NOP
11354  056424                  15$:
11355
11356                          ;REPORT AN EPROR IF "UNS" IS SET AND "DVC" IS 0
11357  056424  032737  040000  001342  BIT     #UNS,RMER1I     ;WAS UNSAFE ON??
11358  056432  001430                  BEQ     16$             ;NO!!
11359  056434  032737  000200  001370  BIT     #DVC,RMER2I     ;WAS THERE A DEVICE CHECK??
11360  056442  001024                  BNE     16$             ;YES - DON'T REPORT UNSAFE
11361  056444  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11362  056452  042737  040000  001140  BIC     #UNS,$GDDAT
11363  056460  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11364  056466  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11365  056472  112776  000107  000000  MOVB    #107,@(SP)      ;WRITE ERROR NUMBER
11366  056500  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11367  056504  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11368  056506  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11369  056512  000240                  NOP
11370  056514                  16$:
11371
11372                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11373  056514  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11374  056520  105776  000000          TSTB    @(SP)             ;WAS AN ERROR REPORTED??
11375  056524  001403                  BEQ     17$               ;NO!!
11376  056526  062716  000004          ADD     #4,(SP)           ;YES - AUGMENT SP RETURN
11377  056532  000402                  BR      18$
11378  056534  162716  000004  17$:    SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
11379  056540  000240          18$:    NOP
11380  056542  000207                  RTS     PC                ;STATUS CECK IS COMPLETE
11381
```

```
11382                                           .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
11383                                           ;       BR      ???             RETURN HERE IF NO ERROR
11384                                           ;       NOP                     RETURN HERE TO REPORT AN ERROR
11385                                           ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11386                                           ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11387                                           ;       ???                     RETURN HERE IF NO MORE ERRORS
11388
11389   056544                                  DRVSTS:
11390
11391                                           ;CLEAR USER'S ERROR CALL
11392   056544  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11393   056550  105076  000000                        CLRB    @(SP)           ;CLEAR ERROR CALL
11394   056554  162716  000004                        SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
11395                                           ;REPORT ERROR IF RMCS1 NOT INITIALIZED
11396   056560  013737  001326  001142         4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
11397   056566  042737  173700  001142                 BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
11398   056574  012737  004010  001140                 MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
11399   056602  023737  001140  001142                 CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11400   056610  001443                                BEQ     6$              ;BRANCH IF EQUAL
11401   056612  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11402   056616  112776  000141  000000                MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11403   056624  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11404   056630  004736                                JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11405   056632  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11406   056636  000240                                NOP
11407                                           ;REPORT ERROR IF RMDS NOT INITIALIZED
11408   056640  013737  001340  001142         5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
11409   056646  042737  021101  001142                 BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
11410   056654  012737  010600  001140                 MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
11411   056662  023737  001140  001142                 CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11412   056670  001413                                BEQ     6$              ;BRANCH IF EQUAL
11413   056672  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11414   056676  112776  000142  000000                MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11415   056704  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11416   056710  004736                                JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11417   056712  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11418   056716  000240                                NOP
11419                                           ;REPORT ERROR IF RMER1 NOT INITIALIZED
11420   056720  005037  001140                 6$:     CLR     $GDDAT          ;EXPECT 0'S
11421   056724  013737  001342  001142                 MOV     RMER1I,$BDDAT   ;CHECK RMER1
11422   056732  001413                                BEQ     8$              ;BRANCH IF EQUAL
11423   056734  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11424   056740  112776  000143  000000                MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11425   056746  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11426   056752  004736                                JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11427   056754  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11428   056760  000240                                NOP
11429                                           ;REPORT ERROR IF ATA NOT INITIALIZED
11430   056762  013737  001344  001142         8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
11431   056770  010146                                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11432   056772  010246                                MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11433   056774  013701  001446                        MOV     TSTQUE,R1
11434   057000  116102  000001                        MOVB    1(R1),R2
11435   057004  042702  177400                        BIC     #^CATNMSK,R2
11436   057010  005102                                COM     R2
11437   057012  040237  001142                        BIC     R2,$BDDAT
```

```
11438  057016  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11439  057020  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11440  057022  005737  001142                  TST     $BDDAT          ;IS ATTENTION CLEARED??
11441  057026  001413                          BEQ     9$              ;BRANCH IF ATTENTION CLEARED
11442  057030  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11443  057034  112776  000144  000000          MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11444  057042  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11445  057046  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11446  057050  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11447  057054  000240                          NOP
11448                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
11449  057056  013737  001352  001142  9$:     MOV     RMMR1I,$BDDAT   ;CHECK RMMR
11450  057064  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
11451  057072  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
11452  057100  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11453  057106  001413                          BEQ     11$             ;BRANCH IF ZERO
11454  057110  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11455  057114  112776  000145  000000          MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11456  057122  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11457  057126  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11458  057130  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11459  057134  000240                          NOP
11460                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
11461  057136  013737  001366  001142  11$:     MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
11462  057144  042737  140000  001142          BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
11463  057152  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
11464  057160  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11465  057166  001413                          BEQ     15$             ;BRANCH IF EQUAL
11466  057170  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11467  057174  112776  000146  000000          MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11468  057202  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11469  057206  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11470  057210  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11471  057214  000240                          NOP
11472  057216  005037  001140          15$:    CLR     $GDDAT          ;EXPECT ZEROS
11473                          ;REPORT ERROR IF RMEC2 NOT RESET
11474  057222  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
11475  057230  001413                          BEQ     17$             ;BRANCH IF 0
11476  057232  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11477  057236  112776  000150  000000          MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11478  057244  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11479  057250  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11480  057252  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11481  057256  000240                          NOP
11482                          ;REPORT ERROR IF RMER2 NOT RESET
11483  057260  013737  001370  001142  17$:     MOV     RMER2I,$BDDAT   ;CHECK RMER2
11484  057266  001413                          BEQ     18$             ;BRANCH IF NO ERROR
11485  057270  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11486  057274  112776  000147  000000          MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11487  057302  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11488  057306  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11489  057310  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11490  057314  000240                          NOP
11491  057316                          18$:
11492
11493  057316                          19$:
```

```
11494
11495                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
11496  057316  062716  000004                    ADD     #4,(SP)          ;MOVE SP TO ERROR CALL
11497  057322  105776  000000                    TSTB    @(SP)            ;WAS AN ERROR DETECTED??
11498  057326  001403                            BEQ     21$              ;NO!!
11499  057330  062716  000004                    ADD     #4,(SP)          ;YES - MOVE SP TO ERROR RETURN
11500  057334  000402                            BR      23$
11501  057336  162716  000004           21$:     SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
11502  057342  000240                    23$:     NOP
11503  057344  000207                            RTS     PC               ;RETURN TO USER
```

```
11504                                   .SBTTL  SEARCH STATUS CHECK SUBROUTINE
11505
11506                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEARCH OPERATIONS USING
11507                                   ;STATUS STORED IN THE GET BUFFER AND TEST CONDITIONS STORED IN THE
11508                                   ;PUT BUFFER.
11509
11510                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS
11511                                   ;DETECTED AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN
11512                                   ;THE USER'S "ERROR" TRAP.
11513
11514                                   ;THE FOLLOWING CONDITIONS ARE CHECKED:
11515
11516                                   ;       .ANY ERROR WHICH OCCURRED WHILE READING OR WRITING REMOTE
11517                                   ;REGISTERS IS REPORTED, I.E., "MCPE"=1 OR "PAR"=1
11518
11519                                   ;       ."IAE" STATUS IS CHECKED IN ACCORDANCE WITH THE VALUE DETERMINED
11520                                   ;BY THE PROGRAM, WHICH IS BASED ON FORMAT AND ADDRESS.
11521
11522                                   ;       ."OPI", IF SET, IS REPORTED AS 1) "OPI" DUE TO MOL=0, OR 2)
11523                                   ;"OPI" DUE TO ON CYLINDER LATCH.
11524
11525                                   ;       ."IVC", IF SET, IS REPORTED AS 1) "IVC" ERROR WITH VOLUME
11526                                   ;VALID ZERO, OR 2) ERRONEOUS "IVC" ERROR WITH VOLUME VALID SET.
11527
11528                                   ;       ."SKI" IS REPORTED IF SET
11529
11530                                   ;       ."DVC" IS REPORTED IF SET
11531
11532                                   ;       .AN ERROR IS REPORTED IF "MOL"=0, OR "PIP"=1, OR "ATA"=0,
11533                                   ;OR "VV"=0
11534
11535                                   ;CALL:
11536                                   ;(1)    JSR     PC,SCHSTS
11537                                   ;(2)    BR      ??              RETURN HERE IF NO ERROR
11538                                   ;(3)    NOP                     RETURN HERE TO REPORT ERROR
11539                                   ;(4)    ERROR                   SUBROUTINE WILL LOAD ERROR #
11540                                   ;(5)    JSR     PC,@(SP)+       GO BACK FOR MORE CHECKS
11541                                   ;(6)    ??                      RETURN AFTER ALL ERRORS REPORTED
11542
11543   057346                          SCHSTS:
11544
11545                                   ;CLEAR USER'S ERROR CALL
11546   057346  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11547   057352  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
11548   057356  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR BR
11549   057362  005037  060710                CLR     200$            ;CLEAR STATUS FLAGS
11550                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING REMOTE
11551                                   ;REGISTERS, I.E., "PAR"=1 AND "DPE"=0.
11552   057366  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
11553   057374  001431                        BEQ     10$             ;NO!!
11554   057376  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS IT CONTROL BUS ERROR??
11555   057404  001025                        BNE     10$             ;PROBABLY NOT!!
11556
11557                                   ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL
11558   057406  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11559   057414  042737  000010  001140        BIC     #PAR,$GDDAT
```

```
11560  057422  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11561  057430  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11562  057434  112776  000050  000000          MOVB    #50,@(SP)               ;WRITE ERROR NUMBER IN CALL
11563  057442  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11564  057446  004736                          JSR     PC,@(SP)+               ;REPORT ERROR
11565  057450  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11566  057454  000240                          NOP
11567  057456  000430                          BR      15$                     ;SKIP FURTHER ERROR CHECKS
11568  057460                          10$:
11569
11570                                  ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN READING REMOTE
11571                                  ;REGISTERS, I.E., "MCPE"=1.
11572  057460  032737  020000  001326          BIT     #MCPE,RMCS1I            ;WAS PARITY ERROR DETECTED??
11573  057466  001426                          BEQ     20$                     ;NO!!
11574
11575                                  ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL.
11576  057470  013737  001326  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
11577  057476  042737  020000  001140          BIC     #MCPE,$GDDAT
11578  057504  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
11579  057512  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11580  057516  112776  000013  000000          MOVB    #13,@(SP)               ;WRITE ERROR NUMBER
11581  057524  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11582  057530  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11583  057532  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11584  057536  000240                          NOP
11585  057540  000137  060662          15$:    JMP     150$                    ;OMIT STATUS CHECKING
11586
11587  057544                          20$:
11588
11589                                  ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK, SECTOR AND CYLINDER
11590                                  ;ADDRESS USED DURING SEARCH.  ALSO SET "SKI" FLAG IF CYLINDER ADDRESS
11591                                  ;IS TOO LARGE
11592  057544  012737  002000  001140          MOV     #IAE,$GDDAT             ;SETUP FOR IAE=1
11593  057552  052737  040000  060710          BIS     #SKI,200$               ;SETUP FOR SKI=1
11594  057560  023727  001432  001466          CMP     RMDC0,#822.             ;CYLINDER > 822??
11595  057566  101025                          BHI     30$                     ;YES - CYLINDER IS INVALID
11596  057570  042737  040000  060710          BIC     #SKI,200$               ;"SKI" SHOULD BE ZERO
11597  057576  123727  001404  000037          CMPB    RMDA0,#31.              ;SECTOR > 31.??
11598  057604  101016                          BHI     30$                     ;YES - SECTOR IS INVALID
11599  057606  123727  001404  000035          CMPB    RMDA0,#29.              ;SECTOR > 29??
11600  057614  101404                          BLOS    25$                     ;NO!!
11601  057616  032737  010000  001360          BIT     #FMT16,RMOFI            ;18 BIT FORMAT??
11602  057624  001406                          BEQ     30$                     ;YES - SECTOR IS INVALID
11603  057626  123727  000007  000004  25$:    CMPB    RMDA+1,#4               ;TRACK > 4??
11604  057634  101002                          BHI     30$                     ;YES - INVALID TRACK
11605  057636  005037  001140                  CLR     $GDDAT                  ;IAE SHOULD BE ZERO
11606  057642                          30$:
11607
11608                                  ;COMPARE EXPECTED AND RECIVED "IAE" STATUS
11609  057642  013737  001342  001142          MOV     RMER1I,$BDDAT           ;GET RECEIVED IAE
11610  057650  042737  175777  001142          BIC     #^C<IAE>,$BDDAT
11611  057656  023737  001140  001142          CMP     $GDDAT,$BDDAT           ;IS IAE CORRECT??
11612  057664  001004                          BNE     40$                     ;NO!!
11613  057666  042737  040000  060710          BIC     #SKI,200$               ;SKI SHOULD BE ZERO
11614  057674  000413                          BR      50$
11615  057676                          40$:
```

```
11616
11617                                   ;REPORT INCORRECT IAE STATUS VIA USER'S ERROR CALL
11618  057676  062716  000004                    ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11619  057702  112776  000257  000000            MOVB    #257,@(SP)              ;WRITE ERROR NUMBER IN CALL
11620  057710  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11621  057714  004736                            JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11622  057716  162716  000010                    SUB     #10,(SP)                ;RETURN SP TO NO ERROR
11623  057722  000240                            NOP
11624  057724                          50$:
11625
11626                                   ;SEE IF "SKI" OR "DVC" OR "IVC" IS SET
11627  057724  032737  050200  001370            BIT     #SKI!DVC!IVC,RMER2I     ;ARE ANY BITS SET??
11628  057732  001531                            BEQ     90$                     ;NO!!
11629
11630                                   ;REPORT ERROR IF "SKI" IS SET AND "SKI" FLAG IS NOT SET
11631  057734  013737  001370  001142            MOV     RMER2I,$BDDAT   ;RECEIVED "SKI" STATUS
11632  057742  042737  137777  001142            BIC     #^CSKI,$BDDAT
11633  057750  013737  060710  001140            MOV     200$,$GDDAT     ;EXPECTED "SKI" STATUS
11634  057756  042737  137777  001140            BIC     #^CSKI,$GDDAT
11635  057764  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;IS "SKI" OK??
11636  057772  001422                            BEQ     60$             ;YES-CHANGE ERROR NUMBER
11637  057774  062716  000004                    ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11638  060000  112776  000263  000000            MOVB    #263,@(SP)              ;WRITE ERROR NUMBER
11639  060006  032737  040000  001140            BIT     #SKI,$GDDAT     ;SHOUL "SKI" BE SET??
11640  060014  001403                            BEQ     55$             ;NO!!
11641  060016  112776  000267  000000            MOVB    #267,@(SP)      ;YES-CHANGE ERROR NUMBER
11642  060024  162716  000002          55$:      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11643  060030  004736                            JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11644  060032  162716  000010                    SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11645  060036  000240                            NOP
11646  060040                          60$:
11647
11648                                   ;REPORT "IVC" ERROR AS
11649                                   ;       .IVC DUE TO LOSS OF VOLUME VALID
11650                                   ;       .ERRONEOUS IVC WITH VOLUME VALID SET
11651  060040  032737  010000  001370            BIT     #IVC,RMER2I             ;IS IVC SET??
11652  060046  001433                            BEQ     80$                     ;NO!!
11653  060050  013737  001370  001140            MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11654  060056  042737  010000  001140            BIC     #IVC,$GDDAT
11655  060064  013737  001370  001142            MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11656  060072  062716  000004                    ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11657  060076  112776  000264  000000            MOVB    #264,@(SP)              ;WRITE ERROR NUMBER IN CALL
11658  060104  032737  000100  001340            BIT     #VV,RMDSI               ;WAS VOLUME VALID??
11659  060112  001403                            BEQ     70$                     ;NO!!
11660  060114  112776  000265  000000            MOVB    #265,@(SP)              ;YES - CHANGE ERROR NUMBER
11661  060122  162716  000002          70$:      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11662  060126  004736                            JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11663  060130  162716  000010                    SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11664  060134  000240                            NOP
11665  060136                          80$:
11666
11667                                   ;REPORT ANY DEVICE FAULT, I.E., "DVC"=1
11668  060136  032737  000200  001370            BIT     #DVC,RMER2I             ;WAS THERE A DEVICE FAULT??
11669  060144  001424                            BEQ     90$                     ;NO!!
11670  060146  013737  001370  001140            MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11671  060154  042737  000200  001140            BIC     #DVC,$GDDAT
```

```
11672  060162  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11673  060170  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11674  060174  112776  000266  000000          MOVB    #266,@(SP)              ;WRITE ERROR NUMBER IN CALL
11675  060202  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11676  060206  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11677  060210  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11678  060214  000240                          NOP
11679  060216                          90$:
11680
11681                                  ;REPORT ANY "OPI" ERROR AS
11682                                  ;       ."OPI" BECAUSE MEDIUM IS NOT ONLINE
11683                                  ;       ."OPI" BECAUSE "ON CYLINDER" DIDN'T DROP
11684  060216  032737  020000  001342          BIT     #OPI,RMER1I             ;WAS OPI SET??
11685  060224  001433                          BEQ     110$                    ;NO!!
11686  060226  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11687  060234  042737  020000  001140          BIC     #OPI,$GDDAT
11688  060242  013737  00134?  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11689  060250  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11690  060254  112776  000270  000000          MOVB    #270,@(SP)              ;SETUP ERROR FOR MOL=0
11691  060262  032737  010000  001340          BIT     #MOL,RMDSI              ;WAS MOL 0??
11692  060270  001403                          BEQ     105$                    ;YES!!
11693  060272  112776  000271  000000          MOVB    #271,@(SP)              ;MOL WAS 1 - CHANGE ERROR
11694  060300  162716  000002          105$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11695  060304  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11696  060306  162716  000010                  SUB     #10,(SP)                ;RESTORE  P TO NO ERROR
11697  060312  000240                          NOP
11698  060314                          110$:
11699
11700                                  ;SEE IF "ATA"="MOL"="VV"=1 AND "PIP"=0
11701  060314  013746  001340                  MOV     RMDSI,-(SP)             ;GET DRIVE STATUS
11702  060320  042716  047677                  BIC     #^C<ATA!PIP!MOL!VV>,(SP)
11703  060324  022726  110100                  CMP     #ATA!MOL!VV,(SP)+       ;IS DRIVE STATUS CORRECT??
11704  060330  001554                          BEQ     150$                    ;YES!!
11705
11706                                  ;REPORT AN ERROR IF MOL=0 AND OPI ERROR WAS NOT REPORTED, I.E., OPI=0
11707  060332  032737  010000  001340          BIT     #MOL,RMDSI              ;WAS MEDIUM OFF LINE??
11708  060340  001030                          BNE     120$                    ;NO!!
11709  060342  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11710  060350  052737  010000  001140          BIS     #MOL,$GDDAT
11711  060356  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11712  060364  032737  020000  001342          BIT     #OPI,RMER1I             ;WAS OPI REPORTED BEFORE??
11713  060372  001013                          BNE     120$                    ;YES - DON'T REPORT MOL=0
11714  060374  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11715  060400  112776  000272  000000          MOVB    #272,@(SP)              ;WRITE ERROR NUMBER IN CALL
11716  060406  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11717  060412  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11718  060414  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11719  060420  000240                          NOP
11720  060422                          120$:
11721
11722                                  ;REPORT AN ERROR IF PIP IS STIL SET AND SKI IS RESET
11723  060422  032737  020000  001340          BIT     #PIP,RMDSI              ;IS POSITIONING IN PROGRESS??
11724  060430  001430                          BEQ     130$                    ;NO!!
11725  060432  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11726  060440  001024                          BNE     130$                    ;YES-DONT REPORT PIP
11727  060442  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
```

```
11728  060450  042737  020000  001140          BIC     #PIP,$GDDAT
11729  060456  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11730  060464  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11731  060470  112776  000273  000000          MOVB    #273,@(SP)              ;WRITE ERROR NUMBER IN CALL
11732  060476  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11733  060502  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11734  060504  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11735  060510  000240                          NOP
11736  060512                          130$:
11737
11738                          ;REPORT AN ERROR IF VOLUME IS NOT VALID AND IVC=0
11739  060512  032737  000100  001340          BIT     #VV,RMDSI               ;IS VOLUME VALID??
11740  060520  001030                          BNE     140$                    ;YES!!
11741  060522  032737  010000  001370          BIT     #IVC,RMER2I             ;WAS IVC DETECTED??
11742  060530  001024                          BNE     140$                    ;YES - DON'T REPORT VV=0
11743  060532  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11744  060540  052737  000100  001140          BIS     #VV,$GDDAT
11745  060546  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11746  060554  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USERS ERROR CALL
11747  060560  112776  000350  000000          MOVB    #350,@(SP)              ;WRITE ERROR NUMBER IN CALL
11748  060566  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11749  060572  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11750  060574  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11751  060600  000240                          NOP
11752  060602                          140$:
11753
11754                          ;REPORT AN ERROR IF ATTENTION IS NOT SET
11755  060602  032737  100000  001340          BIT     #ATA,RMDSI              ;IS ATA ON??
11756  060610  001024                          BNE     150$                    ;YES!!
11757  060612  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11758  060620  052737  100000  001140          BIS     #ATA,$GDDAT
11759  060626  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11760  060634  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11761  060640  112776  000351  000000          MOVB    #351,@(SP)              ;WRITE ERROR NUMBER IN CALL
11762  060646  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11763  060652  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11764  060654  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11765  060660  000240                          NOP
11766  060662                          150$:
11767
11768                          ;AUGMENT THE RETURN ADDRESS IF AN ERROR WAS DETECTED
11769  060662  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11770  060666  105776  000000                  TSTB    @(SP)                   ;WAS ERROR FOUND??
11771  060672  001403                          BEQ     160$                    ;NO!!
11772  060674  062716  000004                  ADD     #4,(SP)                 ;YES - CHANGE RETURN
11773  060700  000402                          BR      170$
11774  060702  162716  000004          160$:   SUB     #4,(SP)                 ;NO ERROR FOUND
11775  060706  000207          170$:   RTS     PC                      ;RETURN TO USER
11776
11777  060710  000000          200$:   .WORD                           ;STORAGE FOR FLAGS
```

```
                                .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
                                ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
                                ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
                                ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.

                                ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
                                ;IF TRUE:

                                ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
                                ;THAT MOL IS ASSUMED TO HAVE BEEN SET
                                ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
                                ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
                                ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
                                ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT

                                ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.

                                ;(1)    JSR     PC,STCDRVSTS
                                ;       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

060712                          STCDRVSTS:

                                ;CLEAR USER'S ERROR CALL
060712  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
060716  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
060722  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
                                ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
060726  013746  001340                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
060732  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
060736  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
060742  001524                          BEQ     30$             ;YES!!

                                ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
060744  032737  010000  001340          BIT     #MOL,RMDSI      ;IS MOL ON ??
060752  001030                          BNE     10$             ;YES!!
060754  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
060762  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
060764  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
060772  052737  010000  001140          BIS     #MOL,$GDDAT
061000  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
061006  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
061012  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
061020  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
061024  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
061026  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
11829  061032  000240                          NOP
11830  061034                                  10$:
11831
11832                                          ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11833  061034  032737  000100  001340          BIT     #VV,RMDSI       ;IS "VV" = 0??
11834  061042  001030                          BNE     20$             ;NO!!
11835  061044  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
11836  061052  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
11837  061054  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11838  061062  052737  000100  001340          BIS     #VV,RMDSI
11839  061070  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11840  061076  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11841  061102  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
11842  061110  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11843  061114  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11844  061116  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11845  061122  000240                          NOP
11846  061124                                  20$:
11847
11848                                          ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11849  061124  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11850  061132  001430                          BEQ     30$             ;NO!!
11851  061134  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11852  061142  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
11853  061144  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11854  061152  042737  020000  001140          BIC     #PIP,$GDDAT
11855  061160  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11856  061166  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11857  061172  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11858  061200  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11859  061204  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11860  061206  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11861  061212  000240                          NOP
11862  061214                                  30$:
11863
11864                                          ;SEE IF "SKI" = "DVC" = 0
11865  061214  013746  001370                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
11866  061220  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
11867  061224  001460                          BEQ     60$             ;BRANCH IF NO ERROR
11868  061226                                  40$:
11869
11870                                          ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11871  061226  032737  000200  001370          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
11872  061234  001424                          BEQ     50$             ;NO!!
11873  061236  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11874  061244  042737  000200  001140          BIC     #DVC,$GDDAT
11875  061252  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11876  061260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
11877  061264  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11878  061272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11879  061276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11880  061300  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11881  061304  000240                          NOP
11882  061306                                  50$:
11883
11884                                          ;REPORT AN ERROR IF "SKI" = 1
```

```
11885  061306  032737  040000  001370          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
11886  061314  001424                          BEQ     60$             ;NO!!
11887  061316  013737  001370  001140          MOV     RMER2I,$GDOAT   ;EXPECTED STATUS
11888  061324  042737  040000  001140          BIC     #SKI,$GDOAT
11889  061332  013737  001370  001142          MOV     RMER2I,$BDOAT   ;RECEIVED STATUS
11890  061340  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11891  061344  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
11892  061352  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11893  061356  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11894  061360  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11895  061364  000240                          NOP
11896  061366                          60$:
11897
11898                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11899  061366  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11900  061372  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11901  061376  001403                          BEQ     70$             ;NO!!
11902  061400  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
11903  061404  000402                          BR      80$
11904  061406  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
11905  061412  000240          80$:    NOP
11906  061414  000207                          RTS     PC              ;RETURN TO USER
```

```
                                        COMPOSITE ERROR CHECK SUBROUTINE

                                        .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE

                                        ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
                                        ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
                                        ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
                                        ;THE MASKS ARE APPLIED.

                                        ;CALL:
                                        ;(1)    JSR     PC,CMPERRSTS
                                        ;       .WORD                   MASK FOR ERROR REGISTER 1
                                        ;       .WORD                   MASK FOR ERROR REGISTER 2
                                        ;       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:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
                                        ;BE ZERO

061416                                  CMPERRSTS:

                                        ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
061416  013737  001342  001176                MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
061424  047637  000000  001176                BIC     @(SP),$TMP1     ;MASK   RMER1
061432  062716  000002                        ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
061436  013737  001370  001200                MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
061444  047637  000000  001200                BIC     @(SP),$TMP2     ;MASK RMER2

                                        ;CLEAR USER'S ERROR CALL
061452  062716  000006                        ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
061456  105076  000000                        CLRB    @(SP)           ;CLEAR ERROR NUMBER
061462  162716  000004                        SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN

                                        ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
061466  005737  001176                        TST     $TMP1           ;ANY ERRORS TO REPORT??
061472  001420                              BEQ     5$              ;NO !!
061474  013737  001176  001142              MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
061502  005037  001140                      CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
061506  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
061512  112776  000066  000000              MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
061520  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
061524  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
061526  162716  000010                      SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
061532  000240                            NOP
061534                                  5$:

                                        ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
061534  005737  001200                        TST     $TMP2           ;ANY ERRORS IN RMER2?
061540  001420                              BEQ     10$             ;NO!!

061542  013737  001200  001142              MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
061550  005037  001140                      CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
061554  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11963  061560  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11964  061566  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11965  061572  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11966  061576  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11967  061600  000240                          NOP
11968  061602                          10$:
11969
11970                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11971  061602  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11972  061606  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
11973  061612  001403                          BEQ     20$             ;NO!!
11974  061614  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11975  061620  000402                          BR      30$
11976  061622  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11977  061626  000207                  30$:    RTS     PC              ;RETURN TO USER
11978
11979
```

```
 11980                                  .SBTTL  STOP AND SHUTDOWN SUBROUTINES
 11981
 11982  061630                          STOP:
 11983
 11984                                  ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
 11985  061630  012746  000140                 MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
 11986  061634  012746  061642                 MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 11987  061640  000002                         RTI                     ;;POP NEW PC AND PS
 11988  061642                          64$:
 11989  061642  000240                         NOP
 11990                                  ;RAISE PRIORITY TO INHIBIT INTERRUPT
 11991  061644  012746  000300                 MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
 11992  061650  012746  061656                 MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
 11993  061654  000002                         RTI                     ;;POP NEW PC AND PS
 11994  061656                          65$:
 11995
 11996  061656  000207                 10$:    RTS     PC              ;CONTINUE
 11997
 11998
 11999  061660                          SHUT:
 12000  061660  104401  061702                 TYPE    ,100$           ;TYPE THE HALT MESSAGE
 12001  061664  005737  000042                 TST     42              ;RUNNING STANDALONE ??
 12002  061670  001402                         BEQ     10$             ;YES !!
 12003  061672  000137  042034                 JMP     $EOP            ;NO - GO TO END OF PASS
 12004  061676                          10$:
 12005  061676  000137  005322                 JMP     START           ;GO TO START
 12006  061702  042524  052123  053440 100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
 12007  061710  051501  044040  046101
 12008  061716  042524  006504  005012
 12009  061724     000
 12010          061726                  .EVEN
```

```
12011                                   .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
12012
12013                                   ;;*****************************************************************
12014                                   ;*SAVE R0-R5
12015                                   ;*CALL:
12016                                   ;*      SAVREG
12017                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
12018                                   ;*
12019                                   ;*TOP---(+16)
12020                                   ;* +2---(+18)
12021                                   ;* +4---R5
12022                                   ;* +6---R4
12023                                   ;* +8---R3
12024                                   ;*+10---R2
12025                                   ;*+12---R1
12026                                   ;*+14---R0
12027
12028  061726                           $SAVREG:
12029  061726  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12030  061730  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12031  061732  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12032  061734  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12033  061736  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12034  061740  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12035  061742  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
12036  061746  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
12037  061752  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
12038  061756  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
12039  061762  000002                          RTI
12040
12041                                   ;*RESTORE R0-R5
12042                                   ;*CALL:
12043                                   ;*      RESREG
12044  061764                           $RESREG:
12045  061764  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
12046  061770  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
12047  061774  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
12048  062000  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
12049  062004  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12050  062006  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12051  062010  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12052  062012  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12053  062014  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12054  062016  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12055  062020  000002                          RTI
12056                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
12057
12058                                   ;;*****************************************************************
12059                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
12060                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
12061                                   ;*CALL:
12062                                   ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
12063                                   ;*      TYPBN                   ;;TYPE IT
12064
12065  062022  010146                   $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
12066  062024  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
12067  062030  000261                    SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
12068  062032  112737  000060  062074  1$:   MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
12069  062040  006101                    ROL     R1              ;;GET THIS BIT
12070  062042  001406                    BEQ     2$              ;;DONE?
12071  062044  105537  062074            ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
12072  062050  104401  062074            TYPE    .$BIN           ;;GO TYPE THIS BIT
12073  062054  000241                    CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
12074  062056  000765                    BR      1$              ;;GO DO THE NEXT BIT
12075  062060  012601                 2$:   MOV     (SP)+,R1        ;;POP THE STACK INTO R1
12076  062062  016666  000002  000004    MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12077  062070  012616                    MOV     (SP)+,(SP)
12078  062072  000002                    RTI                     ;;RETURN TO USER
12079  062074     000     000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
12080                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
12081
12082                                  ;;*****************************************************************
12083                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
12084                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
12085                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
12086                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
12087                                  ;*REPLACED WITH SPACES.
12088                                  ;*CALL:
12089                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
12090                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
12091
12092  062076                       $TYPDS:
12093  062076  010046                    MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12094  062100  010146                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12095  062102  010246                    MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12096  062104  010346                    MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12097  062106  010546                    MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12098  062110  012746  020200            MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
12099  062114  016605  000020            MOV     20(SP),R5       ;;GET THE INPUT NUMBER
12100  062120  100004                    BPL     1$              ;;BR IF INPUT IS POS.
12101  062122  005405                    NEG     R5              ;;MAKE THE BINARY NUMBER POS.
12102  062124  112766  000055  000001    MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
12103  062132  005000                 1$:   CLR     R0              ;;ZERO THE CONSTANTS INDEX
12104  062134  012703  062312            MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
12105  062140  112723  000040            MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
12106  062144  005002                 2$:   CLR     R2              ;;CLEAR THE BCD NUMBER
12107  062146  016001  062302            MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
12108  062152  160105                 3$:   SUB     R1,R5           ;;FORM THIS BCD DIGIT
12109  062154  002402                    BLT     4$              ;;BR IF DONE
12110  062156  005202                    INC     R2              ;;INCREASE THE BCD DIGIT BY 1
12111  062160  000774                    BR      3$
12112  062162  060105                 4$:   ADD     R1,R5           ;;ADD BACK THE CONSTANT
12113  062164  005702                    TST     R2              ;;CHECK IF BCD DIGIT=0
12114  062166  001002                    BNE     5$              ;;FALL THROUGH IF 0
12115  062170  105716                    TSTB    (SP)            ;;STILL DOING LEADING 0'S?
12116  062172  100407                    BMI     7$              ;;BR IF YES
12117  062174  106316                 5$:   ASLB    (SP)            ;;MSD?
12118  062176  103003                    BCC     6$              ;;BR IF NO
12119  062200  116663  000001  177777    MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
12120  062206  052702  000060         6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
12121  062212  052702  000040         7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
12122  062216  110223                    MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
12123  062220  005720                          TST     (R0)+           ;;JUST INCREMENTING
12124  062222  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
12125  062226  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
12126  062230  003002                          BGT     8$              ;;GO TO EXIT
12127  062232  010502                          MOV     R5,R2           ;;GET THE LSD
12128  062234  000764                          BR      6$              ;;GO CHANGE TO ASCII
12129  062236  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
12130  062240  100003                          BPL     9$              ;;BR IF NO
12131  062242  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
12132  062250  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
12133  062252  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12134  062254  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12135  062256  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12136  062260  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12137  062262  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12138  062264  104401  062312                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
12139  062270  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12140  062276  012616                          MOV     (SP)+,(SP)
12141  062300  000002                          RTI                     ;;RETURN TO USER
12142  062302  023420                  $DTBL:  10000.
12143  062304  001750                          1000.
12144  062306  000144                          100.
12145  062310  000012                          10.
12146  062312  000004                  $DBLK:  .BLKW   4
12147                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
12148
12149                                  ;;**************************************************************
12150                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
12151                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
12152                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
12153                                  ;*CALL:
12154                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12155                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
12156                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
12157                                  ;*      .BYTE   M               ;;M=1 OR 0
12158                                  ;*                                      ;;1=TYPE LEADING ZEROS
12159                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
12160                                  ;*
12161                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
12162                                  ;*$TYPOS OR $TYPOC
12163                                  ;*CALL:
12164                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12165                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
12166                                  ;*
12167                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
12168                                  ;*CALL:
12169                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12170                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
12171
12172  062322  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
12173  062326  116637  000001  062545          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
12174  062334  112637  062547                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
12175  062340  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
12176  062344  000406                          BR      $TYPON
12177  062346  112737  000001  062545  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
12178  062354  112737  000006  062547          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
12179  062362  112737  000005  062544  STYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
12180  062370  010346                  MOV     R3,-(SP)        ;;SAVE R3
12181  062372  010446                  MOV     R4,-(SP)        ;;SAVE R4
12182  062374  010546                  MOV     R5,-(SP)        ;;SAVE R5
12183  062376  113704  062547          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
12184  062402  005404                  NEG     R4
12185  062404  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
12186  062410  110437  062546          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
12187  062414  113704  062545          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
12188  062420  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
12189  062424  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
12190  062426  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
12191  062430  000404                  BR      3$              ;;GO DO MSB
12192  062432  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
12193  062434  006105                  ROL     R5
12194  062436  006105                  ROL     R5
12195  062440  010503                  MOV     R5,R3
12196  062442  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
12197  062444  105337  062546          DECB    $OMODE          ;;TYPE THIS DIGIT?
12198  062450  100016                  BPL     7$              ;;BR IF NO
12199  062452  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
12200  062456  001002                  BNE     4$              ;;TEST FOR 0
12201  062460  005704                  TST     R4              ;;SUPPRESS THIS 0?
12202  062462  001403                  BEQ     5$              ;;BR IF YES
12203  062464  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
12204  062466  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
12205  062472  052703  000040          BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
12206  062476  110337  062542          MOVB    R3,8$           ;;SAVE FOR TYPING
12207  062502  104401  062542          TYPE    ,8$             ;;GO TYPE THIS DIGIT
12208  062506  105337  062544          DECB    $OCNT           ;;COUNT BY 1
12209  062512  003347                  BGT     2$              ;;BR IF MORE TO DO
12210  062514  002402                  BLT     6$              ;;BR IF DONE
12211  062516  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
12212  062520  000744                  BR      2$              ;;GO DO THE LAST DIGIT
12213  062522  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
12214  062524  012604                  MOV     (SP)+,R4        ;;RESTORE R4
12215  062526  012603                  MOV     (SP)+,R3        ;;RESTORE R3
12216  062530  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
12217  062536  012616                  MOV     (SP)+,(SP)
12218  062540  000002                  RTI                     ;;RETURN
12219  062542     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
12220  062543     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
12221  062544     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
12222  062545     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
12223  062546  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
12224                          .SBTTL  TYPE ROUTINE
12225
12226                          ;;*****************************************************************
12227                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
12228                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
12229                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
12230                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
12231                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
12232                          ;*
12233                          ;*CALL:
12234                          ;*1) USING A TRAP INSTRUCTION
```

```
12235                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
12236                                   ;*OR
12237                                   ;*      TYPE
12238                                   ;*      MESADR
12239                                   ;*
12240
12241  062550  105737  001173           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
12242  062554  100002                           BPL     1$              ;;BR IF YES
12243  062556  000000                           HALT                    ;;HALT HERE IF NO TERMINAL
12244  062560  000430                           BR      3$              ;;LEAVE
12245  062562  010046                   1$:     MOV     R0,-(SP)        ;;SAVE R0
12246  062564  017600  000002                   MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
12247  062570  122737  000001  001242           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
12248  062576  001011                           BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
12249  062600  132737  000100  001243           BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
12250  062606  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
12251  062610  010037  062620                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
12252  062614  004737  065566                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
12253  062620  000000                   61$:    .WORD   0               ;;MESSAGE ADDRESS
12254  062622  132737  000040  001243   62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
12255  062630  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
12256  062632  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
12257  062634  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
12258  062636  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
12259  062640  012600                   60$:    MOV     (SP)+,R0        ;;RESTORE R0
12260  062642  062716  000002           3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
12261  062646  000002                           RTI                     ;;RETURN
12262  062650  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
12263  062654  001430                           BEQ     8$
12264  062656  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
12265  062662  001006                           BNE     5$
12266  062664  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
12267  062666  104401                           TYPE                    ;;TYPE A CR AND LF
12268  062670  001217                           $CRLF
12269  062672  105037  063026                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
12270  062676  000755                           BR      2$              ;;GET NEXT CHARACTER
12271  062700  004737  062762           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
12272  062704  123726  001172           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
12273  062710  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
12274  062712  013746  001170                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
12275                                           ;;AND THE NULL CHAR.
12276  062716  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
12277  062722  002770                           BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
12278  062724  004737  062762                   JSR     PC,$TYPEC       ;;GO TYPE A NULL
12279  062730  105337  063026                   DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
12280  062734  000770                           BR      7$              ;;LOOP
12281
12282                                   ;HORIZONTAL TAB PROCESSOR
12283
12284  062736  112716  000040           8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
12285  062742  004737  062762           9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
12286  062746  132737  000007  063026           BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
12287  062754  001372                           BNE     9$              ;;TAB STOP
12288  062756  005726                           TST     (SP)+           ;;POP SPACE OFF STACK
12289  062760  000724                           BR      2$              ;;GET NEXT CHARACTER
12290  062762  105777  116176           $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
12291  062766  100375                        BPL     $TYPEC
12292  062770  116677  000002  116170        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
12293  062776  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
12294  063004  001003                        BNE     1$              ;;BRANCH IF NO
12295  063006  105037  063026                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
12296  063012  000406                        BR      $TYPEX          ;;EXIT
12297  063014  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
12298  063022  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
12299  063024  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
12300  063026  000000              $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
12301  063030  000207              $TYPEX: RTS     PC
12302
12303                                .SBTTL  SCOPE HANDLER ROUTINE
12304
12305                                ;;**************************************************************
12306                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
12307                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
12308                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
12309                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
12310                                ;*SW14=1        LOOP ON TEST
12311                                ;*SW11=1        INHIBIT ITERATIONS
12312                                ;*SW09=1        LOOP ON ERROR
12313                                ;*SW08=1        LOOP ON TEST IN SWR<7:0>
12314                                ;*CALL
12315                                ;*      SCOPE           ;;SCOPE=IOT
12316
12317  063032                        $SCOPE:
12318  063032  104410                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
12319  063034  004737  061630                JSR     PC,STOP
12320  063040  032777  040000  116106  1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
12321  063046  001131                        BNE     $OVER           ;;YES IF SW14=1
12322                                ;#####START OF CODE FOR THE XOR TESTER#####
12323  063050  000416              $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
12324                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
12325  063052  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
12326  063056  012737  063076  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
12327  063064  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
12328  063070  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
12329  063074  000500                        BR      $SVLAD          ;;GO TO THE NEXT TEST
12330  063076  022626              5$:       CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
12331  063100  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
12332  063104  000440                        BR      7$              ;;LOOP ON THE PRESENT TEST
12333  063106                        6$:;#####END OF CODE FOR THE XOR TESTER#####
12334  063106  032777  000400  116040        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
12335  063114  001421                        BEQ     2$              ;;BR IF NO
12336  063116  005046                        CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
12337  063120  117716  116030                MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
12338  063124  001414                        BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
12339  063126  022716  000072                CMP     #72,(SP)        ;;CHECK THE NUMBER IN THE SWR
12340  063132  002411                        BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
12341  063134  011637  001116                MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
12342  063140  005316                        DEC     (SP)            ;;BACKUP BY ONE
12343  063142  006316                        ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
12344  063144  062716  063350                ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
12345  063150  013637  001122                MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
12346  063154  000466                        BR      $OVER           ;;GO LOOP ON THE TEST
```

```
 12347  063156  005726                    8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
 12348  063160  105737  001117            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 12349  063164  001421                            BEQ     3$              ;;BR IF NO
 12350  063166  123737  001131  001117            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 12351  063174  101015                            BHI     3$              ;;BR IF NO
 12352  063176  032777  001000  115750            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 12353  063204  001404                            BEQ     4$              ;;BR IF NO
 12354  063206  013737  001124  001122    7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 12355  063214  000446                            BR      $OVER
 12356  063216  105037  001117            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 12357  063222  005037  001206                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 12358  063226  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
 12359  063230  032777  004000  115716    3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
 12360  063236  001011                            BNE     1$              ;;BR IF YES
 12361  063240  005737  001230                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
 12362  063244  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
 12363  063246  005237  001120                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
 12364  063252  023737  001206  001120            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 12365  063260  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 12366  063262  012737  000001  001120    1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 12367  063270  013737  063346  001206            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 12368  063276  105237  001116            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 12369  063302  113737  001116  001226            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 12370  063310  011637  001122                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 12371  063314  011637  001124                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 12372  063320  005037  001210                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 12373  063324  112737  000001  001131            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 12374  063332  013777  001116  115616    $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 12375  063340  013716  001122                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 12376  063344  000002                            RTI                     ;;FIXES PS
 12377  063346  000012                    $MXCNT: 10.                     ;;MAX. NUMBER OF ITERATIONS
 12378  063350                            $SW08TBL:
 12379  063350  007006                            .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
 12380  063352  007204                            .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
 12381  063354  007370                            .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
 12382  063356  007532                            .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
 12383  063360  010416                            .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
 12384  063362  010754                            .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
 12385  063364  011160                            .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
 12386  063366  011504                            .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
 12387  063370  011664                            .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
 12388  063372  012736                            .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
 12389  063374  013236                            .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
 12390  063376  013470                            .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
 12391  063400  014012                            .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
 12392  063402  014336                            .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
 12393  063404  014614                            .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
 12394  063406  015106                            .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
 12395  063410  015356                            .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
 12396  063412  015606                            .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
 12397  063414  016132                            .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
 12398  063416  016552                            .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
 12399  063420  017116                            .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
 12400  063422  017510                            .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
 12401  063424  020164                            .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
 12402  063426  020516                            .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
```

```
12403  063430  021040                    .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
12404  063432  021422                    .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
12405  063434  021740                    .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
12406  063436  022524                    .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
12407  063440  023250                    .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
12408  063442  024040                    .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
12409  063444  024624                    .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
12410  063446  025450                    .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
12411  063450  026006                    .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
12412  063452  026446                    .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
12413  063454  026672                    .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
12414  063456  027234                    .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
12415  063460  027536                    .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
12416  063462  030040                    .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
12417  063464  030366                    .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
12418  063466  030710                    .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
12419  063470  031236                    .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
12420  063472  031564                    .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
12421  063474  032112                    .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
12422  063476  032500                    .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
12423  063500  033114                    .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
12424  063502  033526                    .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
12425  063504  034066                    .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
12426  063506  034430                    .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
12427  063510  034776                    .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
12428  063512  035450                    .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
12429  063514  036126                    .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
12430  063516  036634                    .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
12431  063520  037214                    .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
12432  063522  037602                    .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
12433  063524  040206                    .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
12434  063526  040566                    .WORD   TST70+2         ;;STARTING ADDRESS OF TEST 70
12435  063530  041132                    .WORD   TST71+2         ;;STARTING ADDRESS OF TEST 71
12436  063532  041500                    .WORD   TST72+2         ;;STARTING ADDRESS OF TEST 72
12437                          .SBTTL  ERROR HANDLER ROUTINE
12438
12439                          ;;*****************************************************************
12440                          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
12441                          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
12442                          ;*AND GO TO ERRTYP ON ERROR
12443                          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
12444                          ;*SW15=1        HALT ON ERROR
12445                          ;*SW13=1        INHIBIT ERROR TYPEOUTS
12446                          ;*SW10=1        BELL ON ERROR
12447                          ;*SW09=1        LOOP ON ERROR
12448                          ;*CALL
12449                          ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
12450
12451  063534                  $ERROR:
12452  063534  104410                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12453  063536  105237  001117    7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
12454  063542  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
12455  063544  013777  001116  115404  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
12456  063552  032777  002000  115374  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
12457  063560  001402                  BEQ     1$              ;;NO - SKIP
12458  063562  104401  001212          TYPE    ,$BELL          ;;RING BELL
```

```
12459  063566  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
12460  063572  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
12461  063576  162737  000002  001132          SUB     #2,$ERRPC
12462  063604  117737  115322  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
12463  063612  032777  020000  115334          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
12464  063620  001004                          BNE     20$             ;;SKIP TYPEOUTS
12465  063622  004737  042244                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
12466  063626  104401  001217                  TYPE    ,$CRLF
12467  063632                          20$:
12468  063632  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
12469  063640  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
12470  063642  113737  001130  063654          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
12471  063650  004737  065576                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
12472  063654     000                  21$:    .BYTE   0
12473  063655     000                          .BYTE   0
12474  063656  000777                  22$:    BR      22$             ;;APT ERROR LOOP
12475  063660  005777  115270          2$:     TST     @SWR            ;;HALT ON ERROR
12476  063664  100002                          BPL     3$              ;;SKIP IF CONTINUE
12477  063666  000000                          HALT                    ;;HALT ON ERROR!
12478  063670  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12479  063672  032777  001000  115254  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
12480  063700  001402                          BEQ     4$              ;;BR IF NO
12481  063702  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
12482  063706  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
12483  063712  001402                          BEQ     5$              ;;BR IF NONE
12484  063714  013716  001210                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
12485  063720                          5$:
12486  063720  022737  042224  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
12487  063726  001001                          BNE     6$              ;;BRANCH IF NO
12488  063730  000000                          HALT                    ;;YES
12489  063732                          6$:
12490  063732  000002                          RTI                     ;;RETURN
12491                                  .SBTTL  TTY INPUT ROUTINE
12492
12493                                  ;;**********************************************************************
12494                                  .ENABL  LSB
12495  063734  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
12496  063736  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
12497  063740  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
12498  063742  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
12499          063743                  $TKQEND=.
12500          063744                  .EVEN
12501
12502                                  ;*TK INITIALIZE ROUTINE
12503                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12504                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12505                                  ;
12506                                  ;*CALL:
12507                                  ;*      JSR     PC,$TKINT
12508                                  ;*      RETURN
12509                                  ;
12510  063744  005037  063734          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
12511  063750  012737  063742  063736          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
12512  063756  013737  063736  063740          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12513  063764  012737  064014  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
12514  063772  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
```

```
12515  064000  005777  115156                  TST     @$TKB           ;;CLEAR DONE FLAG
12516  064004  012777  000100  115146          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
12517  064012  000207                          RTS     PC              ;;RETURN TO CALLER
12518
12519                                  ;*TK SERVICE ROUTINE
12520                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12521                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
12522                                  ;*IT IN THE QUEUE.
12523                                  ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
12524                                  ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT)
12525                                  ;
12526  064014  117746  115142          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
12527  064020  042716  177600                  BIC     #↑C177,(SP)     ;;STRIP THE JUNK
12528  064024  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
12529  064030  001007                          BNE     1$              ;;BRANCH IF NO
12530  064032  104401  065130                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
12531  064036  004737  063744                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
12532  064042  005726                          TST     (SP)+           ;;CLEAN UP STACK
12533  064044  000137  061660                  JMP     SHUT            ;;CONTROL C RESTART
12534  064050  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
12535  064054  001004                          BNE     2$              ;;BRANCH IF NO
12536  064056  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
12537  064064  001500                          BEQ     6$              ;;GO TO SWR CHANGE
12538
12539  064066                          2$:
12540  064066  022737  000001  063734          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
12541  064074  001004                          BNE     3$              ;;BRANCH IF NO
12542  064076  104401  001212                  TYPE    ,$BELL          ;;RING THE TTY BELL
12543  064102  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
12544  064104  000451                          BR      5$              ;;EXIT
12545  064106  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
12546  064112  001021                          BNE     32$             ;;BRANCH IF NO
12547  064114  005077  115040                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12548  064120  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
12549  064122  105777  115032          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
12550  064126  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
12551  064130  117746  115026                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
12552  064134  042716  177600                  BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
12553  064140  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
12554  064144  001366                          BNE     31$             ;;BRANCH IF NO
12555  064146  012777  000100  115004          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
12556  064154  000002                          RTI                     ;;RETURN
12557  064156  005237  063734          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
12558  064162  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
12559  064166  002405                          BLT     4$              ;;BRANCH IF YES
12560  064170  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
12561  064174  003002                          BGT     4$              ;;BRANCH IF YES
12562  064176  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
12563  064202  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
12564  064206  005237  063736                  INC     $TKQIN          ;;UPDATE THE POINTER
12565  064212  023727  063736  063743          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
12566  064220  001003                          BNE     5$              ;;BRANCH IF NO
12567  064222  012737  063742  063736          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
12568  064230  000002                  5$:     RTI                     ;;RETURN
12569
12570                                  ;;**************************************************************
```

```
12571                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12572                                   ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
12573                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12574                                   ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12575  064232  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
12576  064240  001124                          BNE     15$             ;;EXIT IF NOT
12577  064242  105777  114712                  TSTB    @$TKS           ;;IS A CHAR WAITING?
12578  064246  100121                          BPL     15$             ;;IF NOT, EXIT
12579  064250  117746  114706                  MOVB    @$TKB,-(SP)     ;;YES
12580  064254  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
12581  064260  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
12582  064264  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
12583                                                                  ;;AND EXIT
12584
12585                                   ;;*****************************************************************
12586                                   ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12587                                   ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12588                                   ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12589  064266  123727  001150  000001  6$:      CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
12590  064274  001674                          BEQ     2$              ;;BRANCH IF YES
12591  064276  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
12592  064300  004737  063744                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
12593  064304  005077  114650                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12594  064310  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
12595
12596  064316  104401  065142                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
12597  064322  104401  065147          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
12598  064326  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
12599  064332  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
12600  064334  104401  065160                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
12601  064340  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
12602  064342  005046                          CLR     -(SP)           ;;THE NEW SWR
12603  064344  105777  114610          7$:     TSTB    @$TKS           ;;CHAR THERE?
12604  064350  100375                          BPL     7$              ;;IF NOT TRY AGAIN
12605
12606  064352  117746  114604                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
12607  064356  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12608
12609  064362  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
12610  064366  001015                          BNE     9$              ;;BRANCH IF NOT
12611  064370  104401  065130                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
12612  064374  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
12613  064400  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
12614  064406  001003                          BNE     8$              ;;BRANCH IF NO
12615  064410  012777  000100  114542          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
12616  064416  000137  061660          8$:     JMP     $RUT            ;;CONTROL-C RESTART
12617
12618
12619  064422  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
12620  064426  001005                          BNE     10$             ;;BRANCH IF NOT
12621  064430  104401  065135                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
12622  064434  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
12623  064440  000737                          BR      19$             ;;LET'S TRY IT AGAIN
12624
12625
12626  064442  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
```

```
12627  064446  001022                        BNE     16$             ;;BRANCH IF NO
12628  064450  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
12629  064454  001403                        BEQ     11$             ;;BRANCH IF YES
12630  064456  016677  000002  114470        MOV     2(SP),@SWR      ;;SAVE NEW SWR
12631  064464  062706  000006          11$:  ADD     #6,SP           ;;CLEAR UP STACK
12632  064470  104401  001217          14$:  TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
12633  064474  123727  001151  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
12634  064502  001003                        BNE     15$             ;;BRANCH IF NOT
12635  064504  012777  000100  114446        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
12636  064512  000002          15$:  RTI                             ;;RETURN
12637  064514  004737  002762          16$:  JSR     PC,$TYPEC       ;;ECHO CHAR
12638  064520  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
12639  064524  002420                        BLT     18$             ;;BRANCH IF YES
12640  064526  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
12641  064532  003015                        BGT     18$             ;;BRANCH IF YES
12642  064534  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
12643  064540  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
12644  064544  001403                        BEQ     17$             ;;BRANCH IF YES
12645  064546  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
12646  064550  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
12647  064552  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
12648  064554  005266  000002          17$:  INC     2(SP)           ;;KEEP COUNT OF CHAR
12649  064560  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
12650  064564  000667                        BR      7$              ;;GET THE NEXT ONE
12651  064566  104401  001216          18$:  TYPE    ,$QUES          ;;TYPE ?<CR><LF>
12652  064572  000720                        BR      20$             ;;SIMULATE CONTROL-U
12653                                .DSABL  LSB
12654
12655
12656                                ;;*****************************************************************
12657                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
12658                                ;*CALL:
12659                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
12660                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
12661                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
12662                                ;
12663
12664  064574  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
12665  064576  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE PS
12666  064604  005066  000004                CLR     4(SP)           ;;GET READY FOR A CHARACTER
12667  064610  005046                        CLR     -(SP)           ;;PUT NEW PS ON STACK
12668  064612  012746  064620                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12669  064616  000002                        RTI                     ;;POP NEW PC AND PS
12670  064620                        64$:
12671  064620  005737  063734          1$:   TST     $TKCNT          ;;WAIT ON A CHARACTER
12672  064624  001775                        BEQ     1$
12673  064626  005337  063734                DEC     $TKCNT          ;;DECREMENT THE COUNTER
12674  064632  117766  177102  000004        MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
12675  064640  005237  063740                INC     $TKQOUT         ;;UPDATE THE POINTER
12676  064644  023727  063740  063743        CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
12677  064652  001003                        BNE     2$              ;;BRANCH IF NO
12678  064654  012737  063742  063740        MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
12679  064662  000002          2$:   RTI                             ;;RETURN
12680                                ;;*****************************************************************
12681                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
12682                                ;*CALL:
```

```
12683                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
12684                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
12685                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
12686
12687  064664  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
12688  064666  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
12689  064670  012703  065120          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
12690  064674  022703  065130          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
12691  064700  101456                          BLOS    4$              ;;BR IF YES
12692  064702  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
12693  064704  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
12694  064706  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
12695  064712  001022                          BNE     5$              ;;BR IF NO
12696  064714  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
12697  064716  001007                          BNE     6$              ;;BR IF NO
12698  064720  112737  000134  065116        MOVB    #'\,9$          ;;TYPE A BACK SLASH
12699  064726  104401  065116                TYPE    ,9$
12700  064732  012716  177777                MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
12701  064736  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
12702  064740  020327  065120                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
12703  064744  103434                          BLO     4$              ;;BR IF YES
12704  064746  111337  065116                MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
12705  064752  104401  065116                TYPE    ,9$             ;;GO TYPE
12706  064756  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
12707  064760  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
12708  064762  001406                          BEQ     7$              ;;BR IF NO
12709  064764  112737  000134  065116        MOVB    #'\,9$          ;;TYPE A BACK SLASH
12710  064772  104401  065116                TYPE    ,9$
12711  064776  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
12712  065000  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
12713  065004  001003                          BNE     8$              ;;BR IF NO
12714  065006  104401  065135                TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
12715  065012  000726                          BR      1$              ;;GO START OVER
12716  065014  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
12717  065020  001011                          BNE     3$              ;;BRANCH IF NO
12718  065022  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
12719  065024  104401  001217                TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
12720  065030  104401  065120                TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
12721  065034  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
12722  065036  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
12723  065042  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
12724  065044  111337  065116          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
12725  065050  104401  065116                TYPE    ,9$
12726  065054  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
12727  065060  001305                          BNE     2$              ;;LOOP IF NOT RETURN
12728  065062  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
12729  065066  104401  001220                TYPE    ,$LF            ;;TYPE A LINE FEED
12730  065072  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
12731  065074  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12732  065076  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
12733  065100  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
12734  065106  012766  065120  000004        MOV     #$TTYIN,4(SP)
12735  065114  000002                          RTI                     ;;RETURN
12736  065116     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
12737  065117     000                          .BYTE   0               ;;TERMINATOR
12738  065120  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
```

```
12739  065130  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>     ;;CONTROL "C"
12740  065135     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
12741  065142  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
12742  065147     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
12743  065154  036440  000040
12744  065160  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
12745  065166  020075     000
12746          065172                  .EVEN
12747                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
12748
12749                                  ;;*****************************************************************
12750                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
12751                                  ;*CHANGE IT TO BINARY.
12752                                  ;*CALL:
12753                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
12754                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
12755                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
12756
12757  065172  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
12758  065174  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
12759  065202  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12760  065204  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12761  065206  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12762  065210  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
12763  065212  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
12764  065214  005001                          CLR     R1              ;;CLEAR DATA WORD
12765  065216  005002                          CLR     R2
12766  065220  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
12767  065222  001412                          BEQ     3$              ;;IF ZERO GET OUT
12768  065224  006301                          ASL     R1              ;;*2
12769  065226  006102                          ROL     R2
12770  065230  006301                          ASL     R1              ;;*4
12771  065232  006102                          ROL     R2
12772  065234  006301                          ASL     R1              ;;*8
12773  065236  006102                          ROL     R2
12774  065240  042716  177770                  BIC     #↑C7,(SP)       ;;STRIP THE ASCII JUNK
12775  065244  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
12776  065246  000764                          BR      2$              ;;LOOP
12777  065250  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
12778  065252  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
12779  065256  010237  065272                  MOV     R2,$HIOCT
12780  065262  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12781  065264  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12782  065266  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12783  065270  000002                          RTI                     ;;RETURN
12784  065272  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
12785                                  .SBTTL  TRAP DECODER
12786
12787                                  ;;*****************************************************************
12788                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12789                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12790                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12791                                  ;*GO TO THAT ROUTINE.
12792
12793  065274  016646  000002          $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
12794  065300  042716  000020                  BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
```

```
12795  065304  012746  065312          1$:     MOV     #1$,-(SP)       ;; T-BIT TRAPS
12796  065310  000002                          RTI                     ;;SET THE NEW STATUS
12797  065312  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
12798  065314  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
12799  065320  005740                          TST     -(R0)           ;;BACKUP BY 2
12800  065322  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
12801  065324  006300                          ASL     R0              ;;POSITION FOR INDEXING
12802  065326  016000  065346                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
12803  065332  000200                          RTS     R0              ;;GO TO ROUTINE
12804
12805
12806                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12807
12808  065334  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
12809  065336  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
12810  065344  000002                          RTI                     ;;RESTORE THE PSW
12811
12812                                  .SBTTL  TRAP TABLE
12813
12814                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12815                                  ;*BY THE "TRAP" INSTRUCTION.
12816
12817                                  ;       ROUTINE
12818                                  ;       -------
12819  065346  065334                  $TRPAD: .WORD   $TRAP2
12820  065350  062550                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
12821  065352  062346                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12822  065354  062322                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12823  065356  062362                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
12824  065360  062076                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12825  065362  062022                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
12826
12827  065364  064322                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
12828
12829  065366  064232                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
12830  065370  064574                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
12831  065372  064664                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
12832  065374  065172                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
12833  065376  061726                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
12834  065400  061764                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
12835                                  .SBTTL  POWER DOWN AND UP ROUTINES
12836
12837                                  ;;**************************************************************
12838                                  ;POWER DOWN ROUTINE
12839  065402  012737  065542  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12840  065410  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12841  065416  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12842  065420  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12843  065422  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12844  065424  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12845  065426  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12846  065430  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12847  065432  017746  113516                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
12848  065436  010637  065546                  MOV     SP,$SAVR6       ;;SAVE SP
12849  065442  012737  065454  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12850  065450  000000                          HALT
```

```
12851  065452  000776                          BR      .-2             ;;HANG UP
12852
12853                                          ;;*****************************************************************
12854                                          ;POWER UP ROUTINE
12855  065454  012737  065542  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12856  065462  013706  065546                  MOV     $SAVR6,SP       ;;GET SP
12857  065466  005037  065546                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
12860  065472  005237  065546          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
12861  065476  001375                          BNE     1$              ;;OF  WORD
12862  065500  012677  113450                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
12863  065504  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12864  065506  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12865  065510  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12866  065512  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12867  065514  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12870  065516  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12871  065520  012737  065402  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12872  065526  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12873  065534  104401                          TYPE                    ;REPORT THE POWER FAILURE
12874  065536  065550                  $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
12875  065540  000002                          RTI
12876  065542  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12877  065544  000776                          BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
12880  065546  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
12881  065550  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
12882  065556  000122
12883                                          .EVEN
12884                                  .SBTTL  APT COMMUNICATIONS ROUTINE
12885                                          ;;*****************************************************************
12886  065560  112737  000001  066024  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
12887  065566  112737  000001  066022  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
12890  065574  000403                          BR      $ATYC
12891  065576  112737  000001  066024  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
12892  065604                          $ATYC:
12893  065604  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12894  065606  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12895  065610  105737  066022                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
12896  065614  001450                          BEQ     5$              ;;IF NOT:  BR
12897  065616  122737  000001  001242          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
12900  065624  001031                          BNE     3$              ;;IF NOT:  BR
12901  065626  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
12902  065634  001425                          BEQ     3$              ;;IF NOT:  BR
12903  065636  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
12904  065642  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12905  065650  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
12906  065654  001375                          BNE     1$              ;;IF NOT:  WAIT
12907  065656  010037  001236                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
12910  065662  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
12911  065664  001376                          BNE     2$
12912  065666  163700  001236                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
12913  065672  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
12914  065674  010037  001240                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
12915  065700  012737  000004  001222          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
12916  065706  000413                          BR      5$
12917  065710  017637  000004  065734  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
```

```
12907  065716  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
12908  065724  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
12909  065730  004737  062550                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
12910  065734  000000                  4$:     .WORD   0
12911  065736                          5$:
12912  065736  105737  066024          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
12913  065742  001416                          BEQ     12$             ;;IF NOT:  BR
12914  065744  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
12915  065750  001413                          BEQ     12$             ;;IF NOT:  BR
12916  065752  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
12917  065756  001375                          BNE     11$             ;;IF NOT:  WAIT
12918  065760  017637  000004  001224          MOV     @4(SP),$FATAL   ;;GET ERROR #
12919  065766  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12920  065774  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
12921  066000  105037  066024          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
12922  066004  105037  066023                  CLRB    $LFLG           ;;CLEAR LOG FLAG
12923  066010  105037  066022                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
12924  066014  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12925  066016  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12926  066020  000207                          RTS     PC              ;;RETURN
12927  066022     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
12928  066023     000          $LFLG:  .BYTE   0               ;;LOG FLAG
12929  066024     000          $FFLG:  .BYTE   C               ;;FATAL FLAG
12930          066026                  .EVEN
12931          000200          APTSIZE=200
12932          000001          APTENV=001
12933          000100          APTSPOOL=100
12934          000040          APTCSUP=040
12935
12936                          .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

066026                          SCTMSG:
066026  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
066110  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
066136     051                  CLSPRN: .ASCII  @)@
066137     075     000          EQUALS: .ASCIZ  @=@
066141     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
066145     077     000          QSTMRK: .ASCIZ  @?@
066147                          HELPQST:
066147     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
066203                          UBUSQST:
066203     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N) ??@
066276  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
066335     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
066362  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
066411     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
066443     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
066473     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
066517     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
066547     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
066603     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
066630                          CNSL07:
066630  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
066706  047040  046525  042502  .ASCII  @ NUMBER(S)@
066720  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@

        066770                  .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.

066770                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
066770  020000  .WORD   OPI                             ;NOP
066772  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (2)
066774  132000  .WORD   ATA!OPI!IVC!IAE                 ;SEEK
066776  130000  .WORD   ATA!OPI!IVC                     ;RECALIBRATE
067000  020000  .WORD   OPI                             ;DRIVE CLEAR
067002  030000  .WORD   OPI!IVC                         ;RELEASE
067004  130000  .WORD   OPI!ATA!IVC                     ;OFFSET
067006  130000  .WORD   OPI!ATA!IVC                     ;RETURN TO CENTERLINE
067010  020000  .WORD   OPI                             ;READ IN PRESET
067012  020000  .WORD   OPI                             ;PACK ACKNOWLEDGE
067014  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (24)
067016  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (26)
067020  132000  .WORD   ATA!OPI!IVC!IAE                 ;SEARCH
067022  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (32)
067024  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (34)
067026  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (36)
067030  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (40)
067032  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (42)
067034  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (44)
067036  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (46)
067040  073300  .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK DATA
067042  073300  .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK HEADER AND DATA
067044  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (54)
067046  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (56)
067050  037200  .WORD   OPI!IVC!WLE!IAE!AOE!HCE         ;WRITE DATA
067052  037000  .WORD   OPI!IVC!WLE!IAE!AOE             ;WRITE HEADER AND DATA
067054  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (64)
067056  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (66)
067060  033300  .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ DATA
067062  033300  .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ HEADER AND DATA
067064  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (74)
067066  130001  .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

                        ATNTBL: .BYTE   1.
                                .BYTE   2.
                                .BYTE   4.
                                .BYTE   8.
                                .BYTE   16.
                                .BYTE   32.
                                .BYTE   64.
                                .BYTE   128.
```

```
067070  001
067071  002
067072  004
067073  010
067074  020
067075  040
067076  100
067077  200
```

```
                        .SBTTL  DATA PATTERN TABLE

                        RGDTPT:
                        MIXED:
067100                          .WORD   0.
067100  000000                  .WORD   0.
067102  000001                  .WORD   1.
067104  000003                  .WORD   3.
067106  000007                  .WORD   7.
067110  000017                  .WORD   15.
067112  000037                  .WORD   31.
067114  000077                  .WORD   63.
067116  000177                  .WORD   127.
067120  000377                  .WORD   255.
067122  000777                  .WORD   511.
067124  001777                  .WORD   1023.
067126  003777                  .WORD   2047.
067130  007777                  .WORD   4095.
067132  017777                  .WORD   8191.
067134  037777                  .WORD   16383.
067136  077777                  .WORD   32767.
067140  177777                  ONES:   .WORD   65535.
067142  177777                  .WORD   65535.
067144  077777                  .WORD   32767.
067146  037777                  .WORD   16383.
067150  017777                  .WORD   8191.
067152  007777                  .WORD   4095.
067154  003777                  .WORD   2047.
067156  001777                  .WORD   1023.
067160  000777                  .WORD   511.
067162  000377                  .WORD   255.
067164  000177                  .WORD   127.
067166  000077                  .WORD   63.
067170  000037                  .WORD   31.
067172  000017                  .WORD   15.
067174  000007                  .WORD   7.
067176  000003                  .WORD   3.
067200  000001                  .WORD   1.
067202  000000                  ZEROS:  .WORD   0.
067204  000000                  .WORD   0.
067206  000001                  .WORD   1.
067210  000002                  .WORD   2.
067212  000004                  .WORD   4.
067214  000010                  .WORD   8.
067216  000020                  .WORD   16.
067220  000040                  .WORD   32.
067222  000100                  .WORD   64.
067224  000200                  .WORD   128.
067226  000400                  .WORD   256.
067230  001000                  .WORD   512.
067232  002000                  .WORD   1024.
067234  004000                  .WORD   2048.
067236  010000                  .WORD   4096.
067240  020000                  .WORD   8192.
067242  040000                  .WORD   16384.
067244  100000                  .WORD   32768.
067246  100000                  .WORD   32768.
```

```
067250  040000                  .WORD   16384.
067252  020000                  .WORD   8192.
067254  010000                  .WORD   4096.
067256  004000                  .WORD   2048.
067260  002000                  .WORD   1024.
067262  001000                  .WORD   512.
067264  000400                  .WORD   256.
067266  000200                  .WORD   128.
067270  000100                  .WORD   64.
067272  000040                  .WORD   32.
067274  000020                  .WORD   16.
067276  000010                  .WORD   8.
067300  000004                  .WORD   4.
067302  000002                  .WORD   2.
067304  000001                  .WORD   1.
067306  000000                  .WORD   0.
067310  177777                  .WORD   65535.
067312  177776                  .WORD   65534.
067314  177774                  .WORD   65532.
067316  177770                  .WORD   65528.
067320  177760                  .WORD   65520.
067322  177740                  .WORD   65504.
067324  177700                  .WORD   65472.
067326  177600                  .WORD   65408.
067330  177400                  .WORD   65280.
067332  177000                  .WORD   65024.
067334  176000                  .WORD   64512.
067336  174000                  .WORD   63488.
067340  170000                  .WORD   61440.
067342  160000                  .WORD   57344.
067344  140000                  .WORD   49152.
067346  100000                  .WORD   32768.
067350  000000                  .WORD   0.
067352  000000                  .WORD   0.
067354  100000                  .WORD   32768.
067356  140000                  .WORD   49152.
067360  160000                  .WORD   57344.
067362  170000                  .WORD   61440.
067364  174000                  .WORD   63488.
067366  176000                  .WORD   64512.
067370  177000                  .WORD   65024.
067372  177400                  .WORD   65280.
067374  177600                  .WORD   65408.
067376  177700                  .WORD   65472.
067400  177740                  .WORD   65504.
067402  177760                  .WORD   65520.
067404  177770                  .WORD   65528.
067406  177774                  .WORD   65532.
067410  177776                  .WORD   65534.
067412  177777                  .WORD   65535.
067414  125252          EARLY:  .WORD   43690.
067416  152525                  .WORD   43690./2
067420  125252                  .WORD   43690.
067422  177777                  .WORD   65535.
067424  177776                  .WORD   65534.
067426  177775                  .WORD   65533.
```

```
067430  177773          .WORD   65531.
067432  177767          .WORD   65527.
067434  177757          .WORD   65519.
067436  177737          .WORD   65503.
067440  177677          .WORD   65471.
067442  177577          .WORD   65407.
067444  177377          .WORD   65279.
067446  176777          .WORD   65023.
067450  175777          .WORD   64511.
067452  173777          .WORD   63487.
067454  167777          .WORD   61439.
067456  157777          .WORD   57343.
067460  137777          .WORD   49151.
067462  077777          .WORD   32767.
067464  077777          .WORD   32767.
067466  137777          .WORD   49151.
067470  157777          .WORD   57343.
067472  167777          .WORD   61439.
067474  173777          .WORD   63487.
067476  175777          .WORD   64511.
067500  176777          .WORD   65023.
067502  177377          .WORD   65279.
067504  177577          .WORD   65407.
067506  177677          .WORD   65471.
067510  177737          .WORD   65503.
067512  177757          .WORD   65519.
067514  177767          .WORD   65527.
067516  177773          .WORD   65531.
067520  177775          .WORD   65533.
067522  177776          .WORD   65534.
067524  177777          .WORD   65535.
067526          ENRGDT:
```

```
                                        .SBTT'  ERROR MESSAGE TABLE

067526  074122  000000          EMT1:   .WORD   EMS1,0
067532  074171  074206  000000  EMT2:   .WORD   EMS2,EMS3,0
067540  074171  074251  000000  EMT3:   .WORD   EMS2,EMS4,0
067546  074314  074344  000000  EMT4:   .WORD   EMS5,EMS6,0
067554  074314  074456  000000  EMT5:   .WORD   EMS5,EMS10,0
067562  101151  076337  000000  EMT6:   .WORD   EMS167,EMS64,0
067570  077105  101176  000000  EMT7:   .WORD   EMS110,EMS170,0
067576  074411  000000          EMT10:  .WORD   EMS7,0
067602  074456  000000          EMT11:  .WORD   EMS10,0
067606  074520  074531  000000  EMT12:  .WORD   EMS11,EMS12,0
067614  074572  074603  074614  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
067626  074666  076337  000000  EMT14:  .WORD   EMS17,EMS64,0
067634  074520  074751  000000  EMT15:  .WORD   EMS11,EMS21,0
067642  074520  074774  075123  EMT16:  .WORD   EMS11,EMS22,EMS27,0
067652  074520  075010  075134  EMT17:  .WORD   EMS11,EMS23,EMS30,0
067662  074520  075036  075134  EMT20:  .WORD   EMS11,EMS24,EMS30,0
067672  074520  075065  075123  EMT21:  .WORD   EMS11,EMS25,EMS27,0
067702  074520  075102  075123  EMT22:  .WORD   EMS11,EMS26,EMS27,0
067712  074520  075144  075134  EMT23:  .WORD   EMS11,EMS31,EMS30,0
067722  074520  075173  075134  EMT24:  .WORD   EMS11,EMS32,EMS30,0
067732  074520  075222  075134  EMT25:  .WORD   EMS11,EMS33,EMS30,0
067742  074520  075250  075134  EMT26:  .WORD   EMS11,EMS34,EMS30,0
067752  074520  075321  075134  EMT27:  .WORD   EMS11,EMS35,EMS30,0
067762  074520  075350  075134  EMT30:  .WORD   EMS11,EMS36,EMS30,0
067772  074520  075377  075134  EMT31:  .WORD   EMS11,EMS37,EMS30,0
070002  074520  075425  075134  EMT32:  .WORD   EMS11,EMS40,EMS30,0
070012  074520  075454  075134  EMT33:  .WORD   EMS11,EMS41,EMS30,0
070022  074520  075502  075134  EMT34:  .WORD   EMS11,EMS42,EMS30,0
070032  074520  075531  075134  EMT35:  .WORD   EMS11,EMS43,EMS30,0
070042  074520  075560  075134  EMT36:  .WORD   EMS11,EMS44,EMS30,0
070052  074520  075633  075134  EMT37:  .WORD   EMS11,EMS45,EMS30,0
070062  076423  074726  000000  EMT40:  .WORD   EMS66,EMS20,0
070070  076611  100317  076617  EMT41:  .WORD   EMS75,EMS141,EMS76,0
070100  100410  100420  076545  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
070112  075725  076043  076617  EMT43:  .WORD   EMS47,EMS53,EMS76,0
070122  076650  076043  076617  EMT44:  .WORD   EMS77,EMS53,EMS76,0
070132  076676  076043  076617  EMT45:  .WORD   EMS100,EMS53,EMS76,0
070142  076724  076043  076617  EMT46:  .WORD   EMS101,EMS53,EMS76,0
070152  076355  076337  000000  EMT47:  .WORD   EMS65,EMS64,0
070160  074572  074614  076311  EMT50:  .WORD   EMS13,EMS15,EMS63,0
070170  074520  075676  075134  EMT51:  .WORD   EMS11,EMS[illegible],EMS30,EMS67,0
070202  075725  076043  076473  EMT52:  .WORD   EMS47,EMS[illegible],EMS67,EMS115,EMS140,EMS14[illegible],0
070220  075725  076043  076473  EMT53:  .WORD   EMS47,EMS[illegible],EMS67,EMS115,EMS141,EMS164,0
070236  075754  076043  076473  EMT54:  .WORD   EMS50,EMS[illegible],EMS67,0
070246  100345  100362  076043  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
070260  076003  076545  076473  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
070276  101151  076555  076473  EMT57:  .WORD   EMS167,EMS73,EMS67,0
070306  076126  076473  077305  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
070324  076532  076126  076473  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
070344  100276  076473  077305  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
070360  000000          EMT63:  .WORD
070362  100503  100546  076520  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
070402  076031  101576  101757  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
070422  101110  077001  076545  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
070434  101110  077001  076545  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
070446  075676  074726  101003  EMT70:  .WORD   EMS46,EMS20,EMS163,0
070456  076532  076724  101003  EMT71:  .WORD   EMS71,EMS101,EMS163,0
070466  075725  076545  101003  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
070504  075725  076545  101003  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
070522  076126  076043  101003  EMT74:  .WORD   EMS56,EMS53,EMS163,0
070532  076532  076126  101003  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
070552  075754  076043  101003  EMT76:  .WORD   EMS50,EMS53,EMS163,0
070562  100345  100362  076043  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
070574  076611  100317  101003  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
070612  076611  100503  101003  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
070630  101151  076[illegible]5  101003  EMT102: .WORD   EMS167,EMS173,EMS163,0
070640  100466  100[illegible]2  076572  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
070652  076003  076572  101003  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
070670  101110  076676  101003  EMT105: .WORD   EMS165,EMS100,EMS163,0
070700  101110  076[illegible]50  101003  EMT106: .WORD   EMS165,EMS77,EMS163,0
070710  100410  100420  076043  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
070730  077105  077145  077312  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
070742  077231  074251  000000  EMT111: .WORD   EMS113,EMS4,0
070750  077231  074206  000000  EMT112: .WORD   EMS113,EMS3,0
070756  077105  077305  077312  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
070772  077231  077364  000000  EMT114: .WORD   EMS113,EMS120,0
071000  077401  100041  100065  EMT115: .WORD   EMS121,EMS132,EMS133,0
071010  077444  100041  100065  EMT116: .WORD   EMS122,EMS132,EMS133,0
071020  077501  100041  100065  EMT117: .WORD   EMS123,EMS132,EMS133,0
071030  077544  100041  100065  EMT120: .WORD   EMS124,EMS132,EMS133,
071040  077576  100041  100065  EMT121: .WORD   EMS125,EMS132,EMS133,0
071050  077641  100041  100065  EMT122: .WORD   EMS126,EMS132,EMS133,0
071060  100241  100041  100065  EMT123: .WORD   EMS127,EMS132,EMS133,0
071070  077743  100041  100065  EMT124: .WORD   EMS130,EMS132,EMS133,0
071100  100001  100041  100065  EMT125: .WORD   EMS131,EMS132,EMS133,0
071110  077401  100041  100110  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
071122  077444  100041  100110  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
071134  077501  100041  100110  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
071146  077544  100041  100110  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
071160  077576  100041  100110  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
071172  077641  100041  100110  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
071204  100241  100041  100110  EMT134: .WORD   EMS127,EMS132,EMS134,EMS123,0
071216  077743  100041  100110  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
071230  100001  100041  100110  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
071242  077401  100041  100152  EMT137: .WORD   EMS121,[illegible],0
071252  077501  100041  100152  EMT140: .WORD   EMS123,[illegible],0
071262  077401  100041  100214  EMT141: .WORD   EMS121,[illegible],0
071272  100241  100041  100214  EMT142: .WORD   EMS127,[illegible],0
071302  077544  100041  100214  EMT143: .WORD   EMS124,[illegible],0
071312  077576  100041  100214  EMT144: .WORD   EMS125,[illegible],0
071322  077641  100041  100214  EMT145: .WORD   EMS126,[illegible],0
071332  100001  100041  100214  EMT146: .WORD   EMS131,[illegible],0
071342  100750  100041  100214  EMT147: .WORD   EMS162,[illegible],0
071352  077743  100041  100214  EMT150: .WORD   EMS130,EMS132,EMS136,0
071362  100276  077305  100317  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
071374  100345  077305  100362  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
071406  100410  100420  100447  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
071424  100410  100420  074726  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
071442  100503  100546  100614  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
071454  100466  100522  100614  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
071466  100466  100522  100650  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
071500  100720  100650  100703  EMT160: .WORD   EMS161,EMS156,EMS160,0
071510  075065  075123  100650  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
071522  075102  075123  100650  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
071534  075725  101003  100276  EMT163: .WORD   EMS47,EMS163,EMS140,0
071544  075725  074726  000000  EMT164: .WORD   EMS47,EMS20,0
071552  076126  074726  000000  EMT165: .WORD   EMS56,EMS20,0
071560  075676  074726  000000  EMT166: .WORD   EMS46,EMS20,0
071566  102277  074726  000000  EMT167: .WORD   EMS224,EMS20,0
071574  101151  100614  101124  EMT170: .WORD   EMS167,EMS154,EMS166,0
071604  101151  100614  100666  EMT171: .WORD   EMS167,EMS154,EMS157,0
071614  101151  100614  100630  EMT172: .WORD   EMS167,EMS154,EMS155,0
071624  101225  100041  100065  EMT173: .WORD   EMS171,EMS132,EMS133,0
071634  101225  100041  100110  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
071646  077231  101400  000000  EMT175: .WORD   EMS113,EMS177,0
071654  101422  101437  000000  EMT176: .WORD   EMS200,EMS201,0
071662  101535  100317  075134  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
071674  101535  076003  075134  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
071706  101535  101550  075134  EMT201: .WORD   EMS203,EMS104,EMS30,EMS202,0
071720  101535  075754  075134  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
071732  101535  100362  075134  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
071744  100503  076572  101505  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
071760  075754  077001  076545  EMT205: .WORD   EMS50,EMS103,EMS72,0
071772  077010  076555  101505  EMT206: .WORD   EMS104,EMS73,EMS202,0
072002  076611  100317  077305  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
072014  076611  100503  000000  EMT210: .WORD   EMS75,EMS150,0
072022  076003  076545  077305  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
072036  100345  076043  077305  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
072052  075754  077001  076043  EMT213: .WORD   EMS50,EMS103,EMS53,0
072062  076031  101576  101124  EMT214: .WORD   EMS52,EMS125,EMS166,EMS206,EMS115,EMS51,EMS72,0
072102  076031  077342  076126  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
072114  076126  075134  076337  EMT216: .WORD   EMS56,EMS30,EMS64,0
072124  075676  075134  076337  EMT217: .WORD   EMS46,EMS30,EMS64,0
072134  075144  075134  076337  EMT220: .WORD   EMS31,EMS30,EMS64,0
072144  075725  075134  076337  EMT221: .WORD   EMS47,EMS30,EMS64,0
072154  076031  077342  076650  EMT222: .WORD   EMS52,EMS117,EMS77,0
072164  076031  077342  076311  EMT223: .WORD   EMS52,EMS117,EMS63,0
072174  000000                  EMT224: .WORD
072176  000000                  EMT225: .WORD
072200  000000                  EMT226: .WORD
072202  000000                  EMT227: .WORD
072204  000000                  EMT230: .WORD
072206  000000                  EMT231: .WORD
072210  000000                  EMT232: .WORD
072212  000000                  EMT233: .WORD
072214  000000                  EMT234: .WORD
072216  000000                  EMT235: .WORD
072220  000000                  EMT236: .WORD
072222  000000                  EMT237: .WORD
072224  000000                  EMT240: .WORD
072226  000000                  EMT241: .WORD
072230  000000                  EMT242: .WORD
072232  000000                  EMT243: .WORD
072234  000000                  EMT244: .WORD
072236  000000                  EMT245: .WORD
072240  101151  100041  101635  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
072250  101151  100041  101662  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
072262  101151  101673  101662  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
072276  101711  101734  000000  EMT251: .WORD   EMS212,EMS213,0
072304  101110  101711  000000  EMT252: .WORD   EMS165,EMS212,0
072312  100465  100522  100650  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
072330  101535  076724  075134  EMT254: .WORD   EMS203,EMS101,EMS30,0
072340  101535  101151  075134  EMT255: .WORD   EMS203,EMS162,EMS30,0
072350  101535  076676  075134  EMT256: .WORD   EMS203,EMS100,EMS30,0
072360  074520  075676  075134  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
072372  076031  077342  076126  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
072404  076031  101576  102101  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
072424  077010  076555  101505  EMT262: .WORD   EMS104,EMS73,EMS202,0
072434  075754  076043  076752  EMT263: .WORD   EMS50,EMS53,EMS102,0
072444  076126  076043  076752  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
072464  076532  076126  076752  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
072504  100345  100362  076043  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
072516  075754  100447  077305  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
072534  075725  076043  076752  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
072550  075725  076043  076752  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
072566  076611  100317  076752  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
072604  076003  076572  076752  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
072622  076311  076043  076203  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
072640  077312  076043  075454  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
072652  075725  076043  076203  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
072666  075725  076043  076203  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
072704  076126  076043  076203  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
072724  076532  076126  076043  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
072746  101110  076676  077001  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
072760  101110  076724  077001  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
072772  101110  076650  077001  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
073004  075676  075134  076337  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
073016  075754  076043  076203  EMT306: .WORD   EMS50,EMS53,EMS57,0
073026  075754  100447  077305  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
073046  100345  100362  076043  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
073060  077010  077001  076043  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
073072  077037  077001  076043  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
073104  100410  100420  077001  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
073124  075502  077001  076043  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
073136  075144  077001  076043  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
073150  076532  075144  077001  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
073162  075531  077001  076203  EMT317: .WORD   EMS43,EMS103,EMS57,0
073172  075633  077001  076203  EMT320: .WORD   EMS45,EMS103,EMS57,0
073202  075660  077001  076203  EMT321: .WORD   EMS44,EMS103,EMS57,0
073212  077066  074726  000000  EMT322: .WORD   EMS106,EMS20,0
073220  075350  077001  076203  EMT323: .WORD   EMS36,EMS103,EMS57,0
073230  101300  075350  077001  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
073242  101260  075350  077001  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
073254  074572  101315  074614  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
073272  100466  100522  076572  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
073304  076423  076043  101323  EMT330: .WORD   EMS66,EMS53,EMS175,0
073314  075222  077001  076043  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
073326  075425  077001  076043  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
073340  076003  076572  076203  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
073356  076611  100317  076203  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
073374  076611  100503  100546  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
073414  076231  076244  076273  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
073424  077312  077342  075250  EMT337: .WORD   EMS116,EMS117,EMS34,0
073434  075250  076043  076055  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
073446  076077  075250  000000  EMT341: .WORD   EMS55,EMS34,0
073454  076031  077342  076126  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
073466  076031  077342  075633  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
073500  076031  077342  075560  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
073512  076031  077342  102121  EMT345: .WORD   EMS52,EMS117,EMS221,0
073522  077066  074726  077305  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
073536  076031  101576  102171  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
073550  076611  100503  076752  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
073566  101151  076555  076752  EMT351: .WORD   EMS167,EMS73,EMS102,0
073576  101775  000000          EMT352: .WORD   EMS215,0
073602  102046  101535  102016  EMT353: .WORD   EMS217,EMS203,EMS216,0
073612  102121  074726  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
073620  102351  103155  103232  EHT1:    .WORD   EH1,STSH1,STSH2,STSH4,0
073632  103155  103232  103357  EHT2:    .WORD   STSH1,STSH2,STSH4,0

073642  102370  000000          EHT110:  .WORD   EH110,0
073646  102377  000000          EHT111:  .WORD   EH111,0

073652  102416  000000          EHT114:  .WORD   EH114,0
073656  102445  103155  103232  EHT223:  .WORD   EH223,STSH1,STSH2,STSH4,0
073670  102473  103155  103232  EHT256:  .WORD   EH256,STSH1,STSH2,STSH4,0

073702  102550  103155  103232  EHT336:  .WORD   EH336,STSH1,STSH2,STSH4,0
073714  102607  103155  103232  EHT337:  .WORD   EH337,STSH1,STSH2,STSH4,0
073726  102746  103155  103232  EHT344:  .WORD   EH344,STSH1,STSH2,STSH4,0

073740  103106  000000          EHT353:  .WORD   EH353,0
```

```
073744  103416  103512  103530  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
073754  103512  103530  103562  EDT2:    .WORD   STSD1,STSD2,STSD4

073762  103424                  EDT110:  .WORD   ED110
073764  103430                  EDT111:  .WORD   ED111

073766  103436                  EDT114:  .WORD   ED114
073770  103446  103512  103530  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4

074000  103456  103512  103530  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
074010  103470  103512  103530  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
074020  103470  103512  103530  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0

074032  103502                  EDT353:  .WORD   ED353
```

```
074034  103575  103613  103613  EFT1:   .WORD   EF111,STSF,STSF,STSF
074044  103613  103613  103613  EFT2:   .WORD   STSF,STSF,STSF

074052  103574                  EFT110: .WORD   EF110
074054  103575                  EFT111: .WORD   EF111

074056  103577                  EFT114: .WORD   EF114
074060  103577  103613  103613  EFT223: .WORD   EF114,STSF,STSF,STSF

074070  103602  103613  103613  EFT336: .WORD   EF336,STSF,STSF,STSF
074100  103602  103613  103613  EFT337: .WORD   EF336,STSF,STSF,STSF
074110  103602  103613  103613  EFT344: .WORD   EF336,STSF,STSF,STSF

074120  103577                  EFT353: .WORD   EF114
```

```
                        .SBTTL  ERROR MESSAGE STRINGS

074122 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
074171    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
074206 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
074251    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
074314 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED @
074344 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
074411    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
074456 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
074520 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
074531    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
074572 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
074603    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
074614 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
074636 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
074666 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
074726 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
074751    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
074774 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
075010 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
075036 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
075065    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
075102 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
075123    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
075134 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
075144 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
075173    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
075222 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
075250 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
075321    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
075350 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
075377    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
075425    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
075454 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
075502 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
075531    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT P) @
075560 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
075633    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
075676 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
075725    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
075754 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
076003    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
076031    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
076043    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
076055    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
076077    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
076126 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
076203    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
076231    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
076244 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
076273    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
076311    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
076337    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
076355    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
076423    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
076473   104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
076520  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
076532  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
076545   111  020123  042523  EMS72:  .ASCIZ  @IS SET @
076555   104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
076572  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
076611   114  051517  020124  EMS75:  .ASCIZ  @LOST @
076617   104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
076650  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
076676  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
076724  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
076752  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
077001   105  051122  051117  EMS103: .ASCIZ  @ERROR @
077010  046042  041522  020042  EMS104: .ASCIZ  @"LSC" (RMER2, BIT 10) @
077037   042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
077066  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
077105   102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
077134  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
077145   127  042510  020116  EMS112: .ASCIZ  @WHEN READING/WRITING RH REGISTERS @
077207   101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
077231   124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
077261   116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
077305   040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
077312  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
077342  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
077364  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
077401   103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
077444  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
077501   103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
077544  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
077576  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
077641   115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
077704  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
077743   105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
100001   115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
100041   116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
100065   125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
100110  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
100152  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
100214  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
100241   104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
100276  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
100317   042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
100345   104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
100362  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
100410  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
100420  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
100447   123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
100466  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
100503   040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
100522  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
100546  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
100572  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
100614  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
100630  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
100650  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
100666  052122  020103  047503  EMS157: .ASCIZ  @RTC COMMAND @
100703     122  050111  041440  EMS160: .ASCIZ  @RIP COMMAND @
100720  043117  051506  052105  EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
100750  051105  047522  020122  EMS162: .ASCIZ  @ERROR REGISTER #2, RMER2, @
101003     104  051125  047111  EMS163: .ASCIZ  @DURING RECALIBRATE @
101027     111  020123  047111  EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
101076  054503  044514  042116          .ASCIZ  @CYLINDER @
101110  047125  054105  042520  EMS165: .ASCIZ  @UNEXPECTED @
101124  042522  040503  044514  EMS166: .ASCIZ  @RECALIBRATE COMMAND @
101151     042  052101  021101  EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
101176  044127  047105  051040  EMS170: .ASCIZ  @WHEN READING REGISTER @
101225     105  051122  051117  EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
101260  047516  051116  041505  EMS172: .ASCIZ  @NONRECOVERABLE @
101330  042522  047503  042526  EMS173: .ASCIZ  @RECOVERABLE @
101315     104  052101  020101  EMS174: .ASCIZ  @DATA @
101323     104  051125  047111  EMS175: .ASCIZ  @DURING WRITE COMMAND @
101351     042  020117  021105  EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
101400  047111  020440  044522  EMS177: .ASCIZ  @IN WRITE PROTECT @
101422  040503  020116  047516  EMS200: .ASCIZ  @CAN NOT SET @
101437     104  040511  047107  EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
101505     104  051125  047111  EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
101535     111  041516  051117  EMS203: .ASCIZ  @INCORRECT @
101550  053442  046122  020042  EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
101576  054105  041505  052125  EMS205: .ASCIZ  @EXECUTED @
101610  044527  044124  041440  EMS206: .ASCIZ  @WITH COMP ERROR SET @
101635     042  047507  020042  EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
101662  051127  052111  047111  EMS210: .ASCIZ  @WRITING @
101673     127  051501  051040  EMS211: .ASCIZ  @WAS RESET BY @
101711     120  047522  051107  EMS212: .ASCIZ  @PROGRAM INTERRUPT @
101734  040527  020123  047516  EMS213: .ASCIZ  @WAS NOT GENERATED @
101757     123  042505  020113  EMS214: .ASCIZ  @SEEK COMMAND @
101775     120  047522  051107  EMS215: .ASCIZ  @PROGRAM TIMEOUT @
102016  052504  044522  043516  EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
102046  047514  045517  040440  EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
102101     123  040505  041522  EMS220: .ASCIZ  @SEARCH COMMAND @
102121     102  042101  051440  EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
102171     101  042040  052101  EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
102222  042510  042101  051105  EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
102277     116  047117  054105  EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
102351    105  050130  052103  EH...:  .ASCIZ  @EXPCTD  RECEVD@
102370 052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
102377    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

102416 042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
102445    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
102473    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
102522 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

102550 042107  042101  051522  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
102607    122  041515  031123  EH337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
102647    137  057537  057537          .ASCII  @++++++++++++++  ++++++++++++++@<CR><LF>
102707    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

102746 046522  051105  020061  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
103007    137  057537  057537          .ASCII  @++++++++++++++   WORD     BIT@<CR><LF>
103046 054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
103106 054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
103126 051040  046115  020101          .ASCIZ  @ RMLA    RMLA    RMOF @

103155    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
103222 020040  051040  040515          .ASCIZ  @    RMAS@
103232 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
103277    040  020040  051040          .ASCIZ  @     RMEC1   RMEC2@
103321    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
103357    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
```

```
                103416                  .EVEN
103416  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
103424  001276  000000          ED110:  .WORD   $BASE,0
103430  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

103436  001354  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
103446  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

103456  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

103470  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
103502  001140  001142  001430  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

103512  001326  001336  001340  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
103530  001330  001332  001334  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
103544  001374  000000          .WORD   RMEC2I,0
103550  001334  001362  001360  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
103562  001352  001366  001354  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
103574  000                     EF110:  .BYTE   0
103575  000  000                EF111:  .BYTE   0,0
103577  000  000  000           EF114:  .BYTE   0,0,0
103602  000  000  000           EF336:  .BYTE   0,0,0,0
103606  000  000  000           EF337:  .BYTE   0,0,0,0,0

103613  000  000  000           STSF:   .BYTE   0,0,0,0,0,0,0
```

```
                                ERROR MESSAGE STRINGS

                                .EVEN
103622  000402                  MFGFIL: .BLKW   258.
104626  177777                          .WORD   -1
104630  177777                          .WORD   -1
104632  000402                  USRFIL:         .BLKW   258.
105636  177777                          .WORD   -1
105640  177777                          .WORD   -1
                                .EVEN
```

```
105642                                  BUFFER:
105642  000402                          BUFONE: .BLKW   258.
106646  000402                          BUFTWO: .BLKW   258.

        105642                          .       =       BUFFER
105642                                  HELP:
105642  005015                          .ASCII  <CR><LF>
105644  046011  051511  020124          .ASCII  @       LIST OF TESTS@<CR><LF>
105664  057411  057537  057537          .ASCII  @       ++++++++++++@<CR><LF>
105704  030524  041411  047117          .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
105737     124  004462  042504          .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
105771     124  004463  051104          .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
106015     124  004464  047125          .ASCII  @T4     UNIBUS INITIALIZE TEST@<CR><LF>
106050  032524  041411  047117          .ASCII  @T5     CONTROLLER CLEAR TEST@<CR><LF>
106102  033124  042411  051122          .ASCII  @T6     ERROR CLEAR TEST@<CR><LF>
106127     124  004467  051104          .ASCII  @T7     DRIVE STATUS TEST@<CR><LF>
106155     124  030061  050011          .ASCII  @T10    PRIMARY/SECONDARY ERROR TEST@<CR><LF>
106217     124  030461  042011          .ASCII  @T11    DIAGNOSTIC MODE TEST@<CR><LF>
106251     124  031061  050011          .ASCII  @T12    PACK ACKNOWLEDGE TEST@<CR><LF>
106304  030524  004463  042522          .ASCII  @T13    RECALIBRATE TEST@<CR><LF>
106332  030524  004464  041101          .ASCII  @T14    ABORT RECALIBRATE TEST@<CR><LF>
106366  030524  004465  053111          .ASCII  @T15    IVC RECALIBRATE TEST@<CR><LF>
106420  030524  004466  040511          .ASCII  @T16    IAE RECALIBRATE TEST@<CR><LF>
106452  030524  004467  042522          .ASCII  @T17    RECALIBRATE AT OFFSET@<CR><LF>
106505     124  030062  042011          .ASCII  @T20    DRIVE CLEAR TEST@<CR><LF>
106533     124  030462  047011          .ASCII  @T21    NOP TEST@<CR><LF>
106551     124  031062  047411          .ASCII  @T22    OFFSET TEST@<CR><LF>
106572  031124  004463  047507          .ASCII  @T23    GO/ATA TEST@<CR><LF>
106613     124  032062  053411          .ASCII  @T24    WRITE ATA TEST@<CR><LF>
106637     124  032462  042411          .ASCII  @T25    ERROR/ATA TEST@<CR><LF>
106663     124  033062  050011          .ASCII  @T26    PROGRAM INTERRUPT TEST@<CR><LF>
106717     124  033462  044411          .ASCII  @T27    INHIBIT INTERRUPT TEST@<CR><LF>
106753     124  030063  051011          .ASCII  @T30    RETURN TO CENTERLINE TEST@<CR><LF>
107012  031524  004461  042522          .ASCII  @T31    READ IN PRESET TEST@<CR><LF>
107043     124  031063  051011          .ASCII  @T32    RMDC CLEAR OFFSET TEST@<CR><LF>
107077     124  031463  044411          .ASCII  @T33    ILLEGAL FUNCTION TEST@<CR><LF>
107132  031524  004464  047111          .ASCII  @T34    INVALID COMMAND TEST@<CR><LF>
107164  031524  004465  047111          .ASCII  @T35    INVALID ADDRESS ERROR TEST@<CR><LF>
107224  031524  004466  051127          .ASCII  @T36    WRITE LOCK ERROR TEST@<CR><LF>
107257     124  033463  047411          .ASCII  @T37    OPI TEST@<CR><LF>
107275     124  030064  042411          .ASCII  @T40    ERROR ABORT TEST@<CR><LF>
107324  032124  004461  046522          .ASCII  @T41    RMR TEST@<CR><LF>
107342  032124  004462  040520          .ASCII  @T42    PARITY ERROR TEST@<CR><LF>
107371     124  031464  044411          .ASCII  @T43    ILLEGAL REGISTER TEST@<CR><LF>
107424  032124  004464  042523          .ASCII  @T44    SEEK LAST CYLINDER@<CR><LF>
107454  032124  004465  042523          .ASCII  @T45    SEEK FIRST CYLINDER@<CR><LF>
107505     124  033064  051411          .ASCII  @T46    SEEK PRIME CYLINDERS@<CR><LF>
107537     124  033464  051411          .ASCII  @T47    SEEK ZERO DIFFERENCE@<CR><LF>
107571     124  030065  051411          .ASCII  @T50    SEEK MAXIMUM DIFFERENCE FORWARD@<CR><LF>
107636  032524  004461  042523          .ASCII  @T51    SEEK ADJACENT FORWARD@<CR><LF>
107671     124  031065  051411          .ASCII  @T52    SEEK ADJACENT REVERSE@<CR><LF>
107724  032524  004463  042523          .ASCII  @T53    SEEK INVALID SECTOR@<CR><LF>
107755     124  032065  051411          .ASCII  @T54    SEEK INVALID TRACK@<CR><LF>
110005     124  032465  051411          .ASCII  @T55    SEEK INVALID CYLINDER@<CR><LF>
110040  032524  004466  053111          .ASCII  @T56    IVC SEEK TEST@<CR><LF>
```

```
110063     124  033465  040411  .ASCII  @T57    ABORT SEEK TEST@<CR><LF>
110110  033124  004460  042523  .ASCII  @T60    SEEK AT OFFSET@<CR><LF>
110134  033124  004461  047514  .ASCII  @T61    LOOK AHEAD TEST@<CR><LF>
110161     124  031066  051411  .ASCII  @T62    SEARCH ON CYLINDER@<CR><LF>
110211     124  031466  051411  .ASCII  @T63    SEARCH OFF CYLINDER@<CR><LF>
110242  033124  004464  042523  .ASCII  @T64    SEARCH INVALID SECTOR@<CR><LF>
110275     124  032466  051411  .ASCII  @T65    SEARCH INVALID TRACK@<CR><LF>
110327     124  033066  051411  .ASCII  @T66    SEARCH INVALID CYLINDER@<CR><LF>
110364  033124  004467  053111  .ASCII  @T67    IVC SEARCH TEST@<CR><LF>
110411     124  030067  040411  .ASCII  @T70    ABORT SEARCH TEST@<CR><LF>
110440  033524  004461  042523  .ASCII  @T71    SEARCH AT OFFSET@<CR><LF>
110466  033524  004462  042510  .ASCII  @T72    HEAD ALIGNMENT SEEK @<CR><LF>
110520  005015                  .ASCII  <CR><LF>
110522  047411  042520  040522  .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
110560  057411  057537  057537  .ASCII  @       ++++++++++++++++++++++++++@<CR><LF>
110616  005015                  .ASCII  <CR><LF>
110620  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
110636  026455  026455  026455  .ASCII  @------                  ----------@<CR><LF>
110674  020040  032461  004411  .ASCII  @  15                    HALT ON ERROR@<CR><LF>
110721     040  030440  004464  .ASCII  @  14                    LOOP ON TEST@<CR><LF>
110745     040  030440  004463  .ASCII  @  13                    INHIBIT ERROR TYPEOUTS@<CR><LF>
111003     040  030440  004462  .ASCII  @  12                    @<CR><LF>
111013     040  030440  004461  .ASCII  @  11                    INHIBIT ITERATIONS@<CR><LF>
111045     040  030440  004460  .ASCII  @  10                    BELL ON ERROR@<CR><LF>
111072  020040  034440  004411  .ASCII  @   9                    LOOP ON ERROR@<CR><LF>
111117     040  020040  004470  .ASCII  @   8                    LOOP ON TEST IN SWR<7:0>@<CR><LF>
111157     040  020040  004467  .ASCII  @   7                    TN128@<CR><LF>
111174  020040  033040  004411  .ASCII  @   6                    TN64@<CR><LF>
111210  020040  032440  004411  .ASCII  @   5                    TN32@<CR><LF>
111224  020040  032040  004411  .ASCII  @   4                    TN16@<CR><LF>
111240  020040  031440  004411  .ASCII  @   3                    TN8@<CR><LF>
111253     040  020040  004462  .ASCII  @   2                    TN4@<CR><LF>
111266  020040  030440  004411  .ASCII  @   1                    TN2@<CR><LF>
111301     040  020040  004460  .ASCII  @   0                    TN1@<CR><LF>
111314  005015                  .ASCII  <CR><LF>
111316  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

DZRMCA.P11 29-JUL-77 13:33 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ABASE = 176700          1945#  2053   2094
ACDW1 = 000000          2053   2096
ACDW2 = 000000          2053   2097
ACKSTS  054206          4876   9364   10975#
ACPUOP= 000000          2053   2068
ADDW0 = 000000          2053   2098
ADDW1 = 000000          2053   2099
ADDW10= 000000          2053
ADDW11= 000000          2053
ADDW12= 000000          2053
ADDW13= 000000          2053
ADDW14= 000000          2053
ADDW15= 000000          2053
ADDW2 = 000000          2053   2100
ADDW3 = 000000          2053   2101
ADDW4 = 000000          2053   2102
ADDW5 = 000000          2053   2103
ADDW6 = 000000          2053   2104
ADDW7 = 000000          2053   2105
ADDW8 = 000000          2053
ADDW9 = 000000          2053
ADEVCT= 000000          2053   2059
ADEVM = 000000          2053   2095
ADR   = 000000          4183#  4184   4225#  4226#  4227   4269#  4270#  4271   4304#  4305#  4307#  4438#  4439#
                        4440   4515#  4516#  4517   4562#  4563#  4564   4628#  4629#  4630   4675#  4676#  4677
                        4822#  4823#  4824   4896#  4897#  4899#  4954#  4955#  4957#  5030#  5031#  5033#  5102#
                        5103#  5105#  5165#  5166#  5168#  5236#  5237#  5238   5299#  5300#  5301   5362#  5363#
                        5364   5437#  5438#  5439   5536#  5537#  5538   5614#  5615#  5616   5697#  5698#  5699
                        5812#  5813#  5814   5896#  5887#  5888   5962#  5963#  5964   6048#  6049#  6050   6117#
                        6118#  6119   6232#  6233#  6234   6341#  6342#  6343   6456#  6457#  6458   6571#  6572#
                        6573   6695#  6696#  6697   6771#  6772#  6773   6865#  6866#  6867   6923#  6924#  6925
                        7004#  7005#  7006#  7077#  7078#  7080#  7150#  7151#  7153#  7230#  7231#  7233#  7310#
                        7311#  7313#  7341#  7392#  7394#  7471#  7472#  7474#  7550#  7551#  7552   7646#  7647#
                        7648   7745#  7746#  7747   7844#  7845#  7846   7923#  7924#  7925   8005#  8006#  8008#
                        8089#  8090#  8091   8190#  8191#  8192   8299#  8300#  8301   8412#  8413#  8414   8508#
                        8509#  8510   8595#  8596#  8597   8689#  8690#  8691   8769#  8770#  8771   8851#  8852#
                        8853   8935#  8936#  8938#
AENV  = 000000          2053   2064
AENVM = 000000          2053   2065
AFATAL= 000000          2053   2056
AMADR1= 000000          2053   2081
AMADR2= 000000          2053   2085
AMADR3= 000000          2053   2088
AMADR4= 000000          2053   2091
AMAMS1= 000000          2053   2075
AMAMS2= 000000          2053   2083
AMAMS3= 000000          2053   2086
AMAMS4= 000000          2053   2089
AMSGAD= 000000          2053   2061
AMSGLG= 000000          2053   2062
AMSGTY= 000000          2053   2055
AMTYP1= 000000          2053   2076
AMTYP2= 000000          2053   2084
AMTYP3= 000000          2053   2087
AMTYP4= 000000          2053   2090
AOE   = 001000          1725#  1736   10145  10160  10175  10190  10194  10441  10445  12936
```

```
APASS = 000000          2053    2058
APE   = 100000          1924#
APRIOR= 000000          2053
APTCSU= 000040         12254   12934#
APTENV= 000001         12247   12468   12890   12932#
APTSIZ= 000200          3960   12931#
APTSP0= 000100         12249   12892   12933#
ARGS  = 000003          4495#   4503#   4657#   4664#   4869#   4876#   4884#   4906#   4929#   4936#   4943#   4965#   5005#
                        5012#   5019#   5040#   5081#   5091#   5112#   5140#   5147#   5154#   5176#   5201#   5203#   5205#
                        5211#   5218#   5225#   5242#   5273#   5280#   5287#   5305#   5327#   5334#   5343#   5350#   5368#
                        5391#   5410#   5419#   5426#   5444#   5472#   5496#   5509#   5518#   5525#   5543#   5588#   5601#
                        5621#   5679#   5686#   5704#   5759#   5778#   5785#   5794#   5801#   5819#   5892#   5915#   5934#
                        5943#   5950#   5969#   5999#   6021#   6030#   6037#   6055#   6077#   6124#   6198#   6239#   6307#
                        6348#   6422#   6463#   6537#   6578#   6661#   6702#   6709#   6712#   6742#   6748#   6749#   6755#
                        6778#   6785#   6814#   6848#   6872#   6881#   6897#   6903#   6930#   6952#   6964#   7015#   7043#
                        7050#   7057#   7066#   7088#   7116#   7123#   7130#   7139#   7162#   7188#   7195#   7202#   7211#
                        7241#   7271#   7278#   7285#   7294#   7321#   7351#   7358#   7365#   7374#   7402#   7431#   7438#
                        7445#   7454#   7482#   7511#   7518#   7525#   7534#   7558#   7588#   7601#   7607#   7614#   7622#
                        7631#   7653#   7683#   7702#   7709#   7717#   7726#   7752#   7782#   7801#   7808#   7816#   7825#
                        7889#   7903#   7912#   7930#   7971#   7978#   7985#   7994#   8016#   8048#   8056#   8063#   8070#
                        8079#   8131#   8196#   8226#   8233#   8253#   8260#   8267#   8276#   8305#   8337#   8344#   8364#
                        8371#   8378#   8387#   8419#   8449#   8462#   8469#   8478#   8514#   8545#   8559#   8568#   8602#
                        8632#   8645#   8652#   8661#   8696#   8736#   8750#   8759#   8776#   8817#   8824#   8831#   8840#
                        8857#   8896#   8903#   8910#   8919#   8958#   8986#   8993#   9000#   9009#
ASNDA   001476          2176#
ASNDC   001474          2175#
ASWREG= 000000          2053    2066
ATA   = 100000          1705#   4584    5403    5406    5407    5465    5467    5469    5502    5505    5564    5567    5568
                        5594    5597    5642    5645    5646    5672    5674    5676    5927    5930    5931    6215    6216
                        6218    6324    6325    6327    6439    6440    6442    6554    6555    6557    6678    6679    6681
                        9972    9973    9975   10707   10708   10743   10746   11238   11239   11278   11281   11702   11703
                       11755   11758   12936
ATESTN= 000000          2053    2057
ATNMSK= 000377          1742#   5571   11435
ATNTBL  067070          4126    4144   12936#
AUNIT = 000000          2053    2060
AUSWR = 000000          2053    2067
AVECT1= 120254          1946#   2053    2092
AVECT2= 000000          2053    2093
A16   = 000400          1895#   6980
A17   = 001000          1894#   6980
BACK  = 000001          4495#   4499    4503#   4507    4657#   4661    4664#   4668    4869#   4873    4876#   4880    4884#
                        4888    4906#   4911    4929#   4933    4936#   4940    4943#   4947    4965#   4970    5005#   5009
                        5012#   5016    5019#   5023    5040#   5045    5081#   5085    5091#   5095    5112#   5117    5140#
                        5144    5147#   5151    5154#   5158    5176#   5181    5201#   5203#   5205#   5209    5211#   5215
                        5218#   5222    5225#   5229    5242#   5247    5273#   5277    5280#   5284    5287#   5291    5305#
                        5310    5327#   5331    5334#   5337#   5340    5343#   5347    5350#   5354    5368#   5373    5391#
                        5395    5410#   5413#   5416    5419#   5423    5426#   5430    5444#   5449    5472#   5475#   5478
                        5496#   5500    5509#   5512#   5515    5518#   5522    5525#   5529    5543#   5548    5588#   5592
                        5601#   5604#   5607    5621#   5626    5679#   5683    5686#   5690    5704#   5709    5759#   5763
                        5778#   5782    5785#   5788#   5791    5794#   5798    5801#   5805    5819#   5824    5892#   5897
                        5915#   5919    5934#   5937#   5940    5943#   5947    5950#   5954    5969#   5974    5999#   6003
                        6021#   6024#   6027    6030#   6034    6037#   6041    6055#   6060    6077#   6081    6124#   6129
                        6198#   6202    6239#   6244    6307#   6311    6348#   6353    6422#   6426    6463#   6468    6537#
                        6541    6578#   6583    6661#   6665    6702#   6707    6709#   6712#   6716    6742#   6746    6748#
                        6743#   6753    6755#   6759    6778#   6783    6785#   6814#   6818    6848#   6852    6872#   6877
```

```
                    6881#  6885   6897#  6901   6903#  6907   6930#  6935   6952#  6956   6964#  6968   7015#
                    7020   7043#  7047   7050#  7054   7057#  7060#  7063   7066#  7070   7088#  7093   7116#
                    7120   7123#  7127   7130#  7133#  7136   7139#  7143   7162#  7167   7188#  7192   7195#
                    7199   7202#  7205#  7208   7211#  7215   7241#  7246   7271#  7275   7278#  7282   7285#
                    7288#  7291   7294#  7298   7321#  7326   7351#  7355   7358#  7362   7365#  7368#  7371
                    7374#  7378   7402#  7407   7431#  7435   7438#  7442   7445#  7448#  7451   7454#  7458
                    7482#  7487   7511#  7515   7518#  7522   7525#  7528#  7531   7534#  7538   7558#  7563
                    7588#  7592   7601#  7605   7607#  7611   7614#  7618   7622#  7625#  7628   7631#  7635
                    7653#  7658   7683#  7687   7702#  7706   7709#  7713   7717#  7720#  7723   7726#  7730
                    7752#  7757   7782#  7786   7801#  7805   7808#  7812   7816#  7819#  7822   7825#  7829
                    7889#  7893   7903#  7906#  7909   7912#  7916   7930#  7935   7971#  7975   7978#  7982
                    7985#  7988#  7991   7994#  7998   8016#  8021   8048#  8056#  8060   8063#  8067   8070#
                    8073#  8076   8079#  8083   8131#  8135   8196#  8201   8226#  8230   8233#  8237   8253#
                    8257   8260#  8264   8267#  8270#  8273   8276#  8290   8305#  8310   8337#  8341   8344#
                    8348   8364#  8368   8371#  8375   8378#  8381#  8384   8387#  8391   8419#  8424   8449#
                    8453   8462#  8466   8469#  8472#  8475   8478#  8482   8514#  8519   8545#  8549   8559#
                    8562#  8565   8568#  8572   8602#  8607   8632#  8636   8645#  8649   8652#  8655#  8658
                    8661#  8665   8696#  8701   8736#  8740   8750#  8753#  8756   8759#  8763   8776#  8781
                    8817#  8821   8824#  8828   8831#  8834#  8837   8840#  8844   8857#  8862   8896#  8900
                    8903#  8907   8910#  8913#  8916   8919#  8923   8958#  8963   8986#  8990   8993#  8997
                    9000#  9003#  9006   9009#  9013
BAI     = 000010    1914#  10361
B800    = 000001    1832#
B801    = 000002    1831#
B802    = 000004    1830#
B803    = 000010    1829#
B804    = 000020    1828#
B805    = 000040    1827#
B806    = 000100    1826#
B807    = 000200    1825#
B808    = 000400    1824#
B809    = 001000    1823#
BIT0    = 000001    1621#  6982   6983   9648
BIT00   = 000001    1611#  1621   1647   1696   1715   1734   1772   1791   1832   1918   1934
BIT01   = 000002    1610#  1620   1646   1695   1733   1771   1790   1831   1917   1933
BIT02   = 000004    1609#  1619   1645   1694   1732   1770   1789   1830   1916   1932
BIT03   = 000010    1608#  1618   1644   1693   1731   1769   1788   1829   1844   1914   1931
BIT04   = 000020    1607#  1617   1643   1692   1730   1787   1828   1913
BIT05   = 000040    1606#  1616   1642   1729   1768   1786   1827   1912
BIT06   = 000100    1605#  1615   1714   1728   1750   1767   1785   1826   1897   1911   1930
BIT07   = 000200    1604#  1614   1713   1727   1749   1766   1784   1808   1825   1843   1896   1910
BIT08   = 000400    1603#  1613   1691   1712   1726   1748   1765   1783   1824   1895   1908   12334
BIT09   = 001000    1602#  1612   1690   1711   1725   1747   1764   1782   1823   1894   1907   12352  12479
BIT1    = 000002    1620#  6982   6983   10378
BIT10   = 002000    1601#  1689   1710   1724   1746   1763   1781   1807   1822   1842   1893   1906   1929
                    12456
BIT11   = 004000    1600#  1641   1709   1723   1762   1780   1798   1806   1821   1841   1905   1928   9376
                    12359
BIT12   = 010000    1599#  1708   1722   1761   1779   1805   1820   1840   1904   1927   9326
BIT13   = 020000    1598#  1707   1721   1760   1778   1797   1819   1839   1891   1903   1926   12463
BIT14   = 040000    1597#  1706   1720   1759   1777   1796   1818   1838   1849   1890   1902   1925   9291
                    12320
BIT15   = 100000    1596#  1705   1719   1758   1776   1795   1817   1837   1848   1889   1901   1924   9278
BIT2    = 000004    1619#  9648
BIT3    = 000010    1618#  9404
BIT4    = 000020    1617#  9354
```

```
BIT5  = 000040          1616#
BIT6  = 000100          1615#   9303
BIT7  = 000200          1614#   3995    9654
BIT8  = 000400          1613#
BIT9  = 001000          1612#
BOTADR  043004          9142*   9160*   9163    9178    9223#
BOTFLG  043006          9128*   9170*   9173    9176*   9224#
BPTVEC= 000014          1628#
BSE   = 100000          1837#  10265
BUFFER  105642          8138    8158   12936#
BUFONE  105642          6173    6282    6397    6512    6636    6729   12936#
BUFTWO  106646         12936#
CC    = 004000          1821#
CH    = 002000          1822#
CHRCNT  043007          9129*   9147*   9153*   9154    9157*   9161    9166*   9177*   9225#
CKSWR = 104410         12318   12452   12478   12829#
CLEAN = ****** U       12936
CLKADR  001500          2177#   9610*   9612    9616*   9618    9643
CLKVCT  001502          2178#   9611*   9617*
CLR   = 000040          1912#  10804
CLRSTS  053326          4495    7588    7683    7782    9313   10840#
CLSPRN  066136          4047    4063    4084   12936#
CMNSTA  006620          4001    4133#
CMPERR  061416          5334    5410    5472    5509    5601    5785    5934    6021    7057    7130    7202    7285    7365
                        7445    7525    7622    7717    7816    7903    7985    8070    8267    8378    8469    8559    8652
                        8750    8831    8910    9000   11927#
CNSL00  066276          4011   12936#
CNSL01  066335          4044   12936#
CNSL02  066362          4053   12936#
CNSL03  066443          4058   12936#
CNSL04  066473          4069   12936#
CNSL05  066547          4079   12936#
CNSL06  066603          4090   12936#
CNSL07  066630          4103   12936#
CNTCLR  053210          4232    4276    4316    4445    4483    4522    4569    4635    4682    4829    6133    6248    6357
                        6472    6587    6712    6881    6944    7576    7671    7770    7851    8096    8486    8581    8674
                        9293   10796#
CONT  = 000100          1785#
CR    = 000015          1536#   4116    9145   12006   12293   12303   12936
CRLF  = 000200          1537#   3984   12264   12303
CYLMSK= 001777          1812#
DBCK  = 100000          1758#
DBEN  = 040000          1759#   4336    4465    6823
DBL   = 002000          1929#
DCK   = 100000          1719#   1736   10303   10323
DDISP = 177570          1543#   2029    3948
DEBL  = 020000          1760#
DEVSEL  051536          6952    9280   10506#
DISPLA  001156          2029#   3948*   3956*  12374*  12455*
DISPRE  000174          1956#   3956
DLT   = 100000          1901#  10147
DMD   = 000001          1772#   1791#   4336    4465    4690    4702    4704    4741    4753    4755    4837    5049    6144
                        6153    6259    6268    6368    6377    6483    6492    6598    6607    6627    6789    6798    6823
                        7859    8705
DPE   = 000010          1844#   5336    5412    5511    5787    5936    6023    7059    7132    7204    7287    7367    7447
                        7527    7624    7719    7818    7905    7987    8072    8269    8380    8471    8561    8654    8752
```

```
                    8833    8912    9002    9809   10089   10535   11124   11554
DPEHI = 040000      1925#
DPELO = 020000      1926#
DPR   = 000400      1712#   4425    4426    4612   10746   10943   10944   11410
DRQ   = 004000      1798#
DRVCLR= 000010      1656#   4798    5250   11398
DRVSTS  056544      5280   11389#
DRY   = 000200      1713#   4425    4426    4612    9776    9872    9875    9876   10746   10943   10944   11410
DSWR  = 177570      1542#   2028    3947
DTE   = 010000      1722#   1736
DTO   = 010000      1761#
DULPRT= 024024      1801#   4291    4294
DVA   = 004000      1641#   4251    4365    9499    9572    9737    9740   10531   10849   11397   11398
DVC   = 000200      1843#   4421    4590    4596    4732    4735    4783    4786    4787   10678   10933   11185   11223
                   11226   11359   11627   11668   11671   11866   11871   11874
EARLY   067414     12936#
EBL   = 020000      1778#
ECH   = 000100      1728#   1736   10286   10301   10319   10325   12936
ECI   = 004000      1806#   5984    6006   10321
ECRC  = 001000      1782#
EDT1    073744      2212    2219    2226    2233    2240    2247    2261    2268    2275    2282    2289    2296    2303
                    2310    2317    2324    2331    2338    2345    2352    2359    2366    2373    2380    2387    2394
                    2401    2408    2415    2422    2429    2436    2443    2450    2457    2464    2471    2478    2485
                    2492    2499    2507    2514    2521    2528    2535    2543    2551    2559    2573    2580    2587
                    2594    2601    2608    2615    2623    2630    2637    2644    2651    2658    2665    2672    2679
                    2686    2693    2700    2707    2749    2756    2763    2770    2777    2784    2791    2798    2805
                    2812    2819    2826    2833    2840    2847    2854    2861    2868    2875    2882    2889    2896
                    2903    2910    2917    2924    2931    2938    2945    2952    2959    2966    2973    2980    2987
                    2994    3001    3008    3023    3030    3037    3044    3058    3065    3072    3079    3086    3093
                    3100    3107    3114    3121    3128    3135    3142    3149    3156    3163    3170    3177    3184
                    3205    3212    3219    3226    3233    3373    3380    3387    3408    3415    3422    3436    3443
                    3450    3457    3464    3471    3478    3485    3493    3500    3508    3515    3522    3530    3538
                    3546    3555    3563    3571    3578    3585    3592    3599    3606    3613    3620    3627    3634
                    3641    3648    3655    3662    3669    3676    3683    3690    3697    3704    3711    3718    3725
                    3732    3739    3746    3753    3760    3767    3802    3809    3823    3830    3837    3844    3851
                    3872   12936#
EDT110  073762      2714   12936#
EDT111  073764      2721    2728   12936#
EDT114  073766      2742   12936#
EDT2    073754      3191    3198    3394    3401   12936#
EDT223  073770      3240    3429   12936#
EDT336  074000      3016    3774    3788    3795   12936#
EDT337  074010      3781   12936#
EDT344  074020      3816   12936#
EDT353  074032      3865   12936#
ED1     103416     12936#
ED110   103424     12936#
ED111   103430     12936#
ED114   103436     12936#
ED223   103446     12936#
ED336   103456     12936#
ED337   103470     12936#
ED353   103502     12936#
EECC  = 000020      1787#
EFT1    074034      2213    2220    2227    2234    2241    2248    2262    2269    2276    2283    2290    2297    2304
                    2311    2318    2325    2332    2339    2346    2353    2360    2367    2374    2381    2388    2395
```

```
                    2402    2409    2416    2423    2430    2437    2444    2451    2458    2465    2472    2479    2486
                    2493    2500    2508    2515    2522    2529    2536    2544    2552    2560    2574    2581    2588
                    2595    2602    2609    2616    2624    2631    2638    2645    2652    2659    2666    2673    2680
                    2687    2694    2701    2708    2750    2757    2764    2771    2778    2785    2792    2799    2806
                    2813    2820    2827    2834    2841    2848    2855    2862    2869    2876    2883    2890    2897
                    2904    2911    2918    2925    2932    2939    2946    2953    2960    2967    2974    2981    2988
                    2995    3002    3009    3024    3031    3038    3045    3059    3066    3073    3080    3087    3094
                    3101    3108    3115    3122    3129    3136    3143    3150    3157    3164    3171    3178    3185
                    3206    3213    3220    3227    3234    3374    3381    3388    3409    3416    3423    3437    3444
                    3451    3458    3465    3472    3479    3486    3494    3501    3509    3516    3523    3531    3539
                    3547    3556    3564    3572    3579    3586    3593    3600    3607    3614    3621    3628    3635
                    3642    3649    3656    3663    3670    3677    3684    3691    3698    3705    3712    3719    3726
                    3733    3740    3747    3754    3761    3768    3803    3810    3824    3831    3838    3845    3852
                    3873   12936#
EFT110  074052      2715   12936#
EFT111  074054      2722    2729   12936#
EFT114  074056      2743   12936#
EFT2    074044      3192    3199    3395    3402   12936#
EFT223  074060      3241    3430   12936#
EFT336  074070      3017    3775    3789    3796   12936#
EFT337  074100      3782   12936#
EFT344  074110      3817   12936#
EFT353  074120      3866   12936#
EF110   103574     12936#
EF111   103575     12936#
EF114   103577     12936#
EF336   103602     12936#
EF337   103606     12936#
EHT1    073620      2211    2218    2225    2232    2239    2246    2260    2267    2274    2281    2288    2295    2302
                    2309    2316    2323    2330    2337    2344    2351    2358    2365    2372    2379    2386    2393
                    2400    2407    2414    2421    2428    2435    2442    2449    2456    2463    2470    2477    2484
                    2491    2498    2506    2513    2520    2527    2534    2542    2550    2558    2572    2579    2586
                    2593    2600    2607    2614    2622    2629    2636    2643    2650    2657    2664    2671    2678
                    2685    2692    2699    2706    2748    2755    2762    2769    2776    2783    2790    2797    2804
                    2811    2818    2825    2832    2839    2846    2853    2860    2867    2874    2881    2888    2895
                    2902    2909    2916    2923    2930    2937    2944    2951    2958    2965    2972    2979    2986
                    2993    3000    3007    3022    3029    3036    3043    3057    3064    3071    3078    3085    3092
                    3099    3106    3113    3120    3127    3134    3141    3148    3155    3162    3169    3176    3183
                    3204    3211    3218    3225    3232    3372    3379    3386    3407    3414    3421    3435    3442
                    3449    3456    3463    3470    3477    3484    3492    3499    3507    3514    3521    3529    3537
                    3545    3554    3562    3570    3577    3584    3591    3598    3605    3612    3619    3626    3633
                    3640    3647    3654    3661    3668    3675    3682    3689    3696    3703    3710    3717    3724
                    3731    3738    3745    3752    3759    3766    3801    3808    3822    3829    3836    3843    3850
                    3871   12936#
EHT110  073642      2713   12936#
EHT111  073646      2720    2727   12936#
EHT114  073652      2741   12936#
EHT2    073632      3190    3197    3393    3400   12936#
EHT223  073656      3239   12936#
EHT256  073670      3428   12936#
EHT336  073702      3015    3773    3787    3794   12936#
EHT337  073714      3780   12936#
EHT344  073726      3815   12936#
EHT353  073740      3864   12936#
EH1     102351     12936#
EH110   102370     12936#
```

```
EH111   102377          12936#
EH114   102416          12936#
EH223   102445          12936#
EH256   102473          12936#
EH336   102550          12936#
EH337   102607          12936#
EH344   102746          12936#
EH353   103106          12936#
EMS1    074122          12936#
EMS10   074456          12936#
EMS100  076676          12936#
EMS101  076724          12936#
EMS102  076752          12936#
EMS103  077001          12936#
EMS104  077010          12936#
EMS105  077037          12936#
EMS106  077066          12936#
EMS11   074520          12936#
EMS110  077105          12936#
EMS111  077134          12936#
EMS112  077145          12936#
EMS113  077231          12936#
EMS114  077261          12936#
EMS115  077305          12936#
EMS116  077312          12936#
EMS117  077342          12936#
EMS12   077531          12936#
EMS120  077364          12936#
EMS121  077401          12936#
EMS122  077444          12936#
EMS123  077501          12936#
EMS124  077544          12936#
EMS125  077576          12936#
EMS126  077641          12936#
EMS127  077704          12936#
EMS13   074572          12936#
EMS130  077743          12936#
EMS131  100001          12936#
EMS132  100041          12936#
EMS133  100065          12936#
EMS134  100110          12936#
EMS135  100152          12936#
EMS136  100214          12936#
EMS137  100241          12936#
EMS14   074603          12936#
EMS140  100276          12936#
EMS141  100317          12936#
EMS142  100345          12936#
EMS143  100362          12936#
EMS144  100410          12936#
EMS145  100420          12936#
EMS146  100447          12936#
EMS147  100466          12936#
EMS15   074614          12936#
EMS150  100503          12936#
EMS151  100522          12936#
```

```
EMS152  100546          12936#
EMS153  100572          12936#
EMS154  100614          12936#
EMS155  100630          12936#
EMS156  100 50          12936#
EMS157  100666          12936#
EMS16   074636          12936#
EMS160  100703          12936#
EMS161  100720          12936#
EMS162  100750          12936#
EMS163  101003          12936#
EMS164  101027          12936#
EMS165  101110          12936#
EMS166  101124          12936#
EMS167  101151          12936#
EMS17   074666          12936#
EMS170  101176          12936#
EMS171  101225          12936#
EMS172  101260          12936#
EMS173  101300          12936#
EMS174  101315          12936#
EMS175  101323          12936#
EMS176  101351          12936#
EMS177  101400          12936#
EMS2    074171          12936#
EMS20   074726          12936#
EMS200  101422          12936#
EMS201  101437          12936#
EMS202  101505          12936#
EMS203  101535          12936#
EMS204  101550          12936#
EMS205  101576          12936#
EMS206  101610          12936#
EMS207  101635          12936#
EMS21   074751          12936#
EMS210  101662          12936#
EMS211  101673          12936#
EMS212  101711          12936#
EMS213  101734          12936#
EMS214  101757          12936#
EMS215  101775          12936#
EMS216  102016          12936#
EMS217  102046          12936#
EMS22   074774          12936#
EMS220  102101          12936#
EMS221  102121          12936#
EMS222  102171          12936#
EMS223  102222          12936#
EMS224  102277          12936#
EMS23   075010          12936#
EMS24   075036          12936#
EMS25   075065          12936#
EMS26   075102          12936#
EMS27   075123          12936#
EMS3    074206          12936#
EMS30   075134          12936#
```

```
EMS31   075144          12936#
EMS32   075173          12936#
EMS33   075222          12936#
EMS34   075250          12936#
EMS35   075321          12936#
EMS36   075350          12936#
EMS37   075377          12936#
EMS4    074251          12936#
EMS40   075426          12936#
EMS41   075454          12936#
EMS42   075502          12936#
EMS43   075531          12936#
EMS44   075560          12936#
EMS45   075633          12936#
EMS46   075676          12936#
EMS47   075725          12936#
EMS5    074314          12936#
EMS50   075754          12936#
EMS51   076003          12936#
EMS52   076031          12936#
EMS53   076043          12936#
EMS54   076055          12936#
EMS55   076077          12936#
EMS56   076126          12936#
EMS57   076203          12936#
EMS6    074344          12936#
EMS60   076231          12936#
EMS61   076244          12936#
EMS62   076273          12936#
EMS63   076311          12936#
EMS64   076337          12936#
EMS65   076355          12936#
EMS66   076423          12936#
EMS67   076473          12936#
EMS7    074411          12936#
EMS70   076520          12936#
EMS71   076532          12936#
EMS72   076545          12936#
EMS73   076555          12936#
EMS74   076572          12936#
EMS75   076611          12936#
EMS76   076617          12936#
EMS77   076650          12936#
EMTVEC= 000030           1631#   3931*   3932*
EMT1    067526           2210   12936#
EMT10   067576           2259   12936#
EMT100  070574           2656   12936#
EMT101  070612           2663   12936#
EMT102  070630           2670   12936#
EMT103  070640           2677   12936#
EMT104  070652           2684   12936#
EMT105  070670           2691   12936#
EMT106  070700           2698   12936#
EMT107  070710           2705   12936#
EMT11   067602           2266   12936#
EMT110  070730           2712   12936#
```

```
EMT111  070742          2719   12936#
EMT112  070750          2726   12936#
EMT113  070756          2733   12936#
EMT114  070772          2740   12936#
EMT115  071000          2747   12936#
EMT116  071010          2754   12936#
EMT117  071020          2761   12936#
EMT12   067606          2273   12936#
EMT120  071030          2768   12936#
EMT121  071040          2775   12936#
EMT122  071050          2782   12936#
EMT123  071060          2789   12936#
EMT124  071070          2796   12936#
EMT125  071100          2803   12936#
EMT126  071110          2810   12936#
EMT127  071122          2817   12936#
EMT13   067614          2280   12936#
EMT130  071134          2824   12936#
EMT131  071146          2831   12936#
EMT132  071160          2838   12936#
EMT133  071172          2845   12936#
EMT134  071204          2852   12936#
EMT135  071216          2859   12936#
EMT136  071230          2866   12936#
EMT137  071242          2873   12936#
EMT14   067626          2287   12936#
EMT140  071252          2880   12936#
EMT141  071262          2887   12936#
EMT142  071272          2894   12936#
EMT143  071302          2901   12936#
EMT144  071312          2908   12936#
EMT145  071322          2915   12936#
EMT146  071332          2922   12936#
EMT147  071342          2929   12936#
EMT15   067634          2294   12936#
EMT150  071352          2936   12936#
EMT151  071362          2943   12936#
EMT152  071374          2950   12936#
EMT153  071406          2957   12936#
EMT154  071424          2964   12936#
EMT155  071442          2971   12936#
EMT156  071454          2978   12936#
EMT157  071466          2985   12936#
EMT16   067642          2301   12936#
EMT160  071500          2992   12936#
EMT161  071510          2999   12936#
EMT162  071522          3006   12936#
EMT163  071534        12936#
EMT164  071544          3021   12936#
EMT165  071552          3028   12936#
EMT166  071560          3035   12936#
EMT167  071566          3042   12936#
EMT17   067652          2308   12936#
EMT170  071574        12936#
EMT171  071604          3056   12936#
EMT172  071614          3063   12936#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
EMT173  071624            3070   12936#
EMT174  071634            3077   12936#
EMT175  071646            3084   12936#
EMT176  071654            3091   12936#
EMT177  071662            3098   12936#
EMT2    067532            2217   12936#
EMT20   067662            2315   12936#
EMT200  071674            3105   12936#
EMT201  071706            3112   12936#
EMT202  071720            3119   12936#
EMT203  071732            3126   12936#
EMT204  071744            3133   12936#
EMT205  071762            3140   12936#
EMT206  071772            3147   12936#
EMT207  072002            3154   12936#
EMT21   067672            2322   12936#
EMT210  072014            3161   12936#
EMT211  072022            3168   12936#
EMT212  072036            3175   12936#
EMT213  072052            3182   12936#
EMT214  072062            3189   12936#
EMT215  072102            3196   12936#
EMT216  072114            3203   12936#
EMT217  072124            3210   12936#
EMT22   067702            2329   12936#
EMT220  072134            3217   12936#
EMT221  072144            3224   12936#
EMT222  072154            3231   12936#
EMT223  072164            3238   12936#
EMT224  072174           12936#
EMT225  072176           12936#
EMT226  072200           12936#
EMT227  072202           12936#
EMT23   067712            2336   12936#
EMT230  072204           12936#
EMT231  072206           12936#
EMT232  072210           12936#
EMT233  072212           12936#
EMT234  072214           12936#
EMT235  072216           12936#
EMT236  072220           12936#
EMT237  072222           12936#
EMT24   067722            2343   12936#
EMT240  072224           12936#
EMT241  072226           12936#
EMT242  072230           12936#
EMT243  072232           12936#
EMT244  072234           12936#
EMT245  072236           12936#
EMT246  072240            3371   12936#
EMT247  072250            3378   12936#
EMT25   067732            2350   12936#
EMT250  072262            3385   12936#
EMT251  072276            3392   12936#
EMT252  072304            3399   12936#
EMT253  072312            3406   12936#
```

EMT254	072330	3413	12936#
EMT255	072340	3420	12936#
EMT256	072350	3427	12936#
EMT2[illegible]	072360	3434	12936#
EMT	067742	2357	12936#
EMT[illegible]0	072372	3441	12936#
EMT261	072404	3448	12936#
EMT262	072424	3455	12936#
EMT263	072434	3462	12936#
EMT264	072444	3469	12936#
EMT265	072464	3476	12936#
EMT266	072504	3483	12936#
EMT267	072516	3491	12936#
EMT27	067752	2364	12936#
EMT270	072534	3498	12936#
EMT271	072550	3506	12936#
EMT272	072566	3513	12936#
EMT273	072604	3520	12936#
EMT274	072622	3528	12936#
EMT275	072640	3536	12936#
EMT276	072652	3544	12936#
EMT277	072666	3553	12936#
EMT3	067540	2224	12936#
EMT30	067762	2371	12936#
EMT300	072704	3561	12936#
EMT301	072724	3569	12936#
EMT302	072746	3576	12936#
EMT303	072760	3583	12936#
EMT304	072772	3590	12936#
EMT305	073004	3597	12936#
EMT306	073016	3604	12936#
EMT307	073026	3611	12936#
EMT31	067772	2378	12936#
EMT310	073046	3618	12936#
EMT311	073060	3625	12936#
EMT312	073072	3632	12936#
EMT313	073104	3639	12936#
EMT314	073124	3646	12936#
EMT315	073136	3653	12936#
EMT316	073150	3660	12936#
EMT317	073162	3667	12936#
EMT32	070002	2385	12936#
EMT320	073172	3674	12936#
EMT321	073202	3681	12936#
EMT322	073212	3688	12936#
EMT323	073220	3695	12936#
EMT324	073230	3702	12936#
EMT325	073242	3709	12936#
EMT326	073254	3716	12936#
EMT327	073272	3723	12936#
EMT33	070012	2392	12936#
EMT330	073304	3730	12936#
EMT331	073314	3737	12936#
EMT332	073326	3744	12936#
EMT333	073340	3751	12936#
EMT334	073356	3758	12936#

```
EMT335  073374            3765   12936#
EMT336  073414            3014    3772   12936#
EMT337  073424            3779   12936#
EMT34   070022            2399   12936#
EMT340  073434            3786   12936#
EMT341  073446            3793   12936#
EMT342  073454            3800   12936#
EMT343  073466            3807   12936#
EMT344  073500            3814   12936#
EMT345  073512            3821   12936#
EMT346  073522            3828   12936#
EMT347  073536            3835   12936#
EMT35   070032            2406   12936#
EMT350  073550            3842   12936#
EMT351  073566            3849   12936#
EMT352  073576            3856   12936#
EMT353  073602            3863   12936#
EMT354  073612            3870   12936#
EMT36   070042            2413   12936#
EMT37   070052            2420   12936#
EMT4    067546            2231   12936#
EMT40   070062            2427   12936#
EMT41   070070            2434   12936#
EMT42   070100            2441   12936#
EMT43   070112            2448   12936#
EMT44   070122            2455   12936#
EMT45   070132            2462   12936#
EMT46   070142            2469   12936#
EMT47   070152            2476   12936#
EMT5    067554            2238   12936#
EMT50   070160            2483   12936#
EMT51   070170            2490   12936#
EMT52   070202            2497   12936#
EMT53   070220            2505   12936#
EMT54   070236            2512   12936#
EMT55   070246            2519   12936#
EMT56   070260            2526   12936#
EMT57   070276            2533   12936#
EMT6    067562            2245   12936#
EMT60   070306            2541   12936#
EMT61   070324            2549   12936#
EMT62   070344            2557   12936#
EMT63   070360          12936#
EMT64   070362            2571   12936#
EMT65   070402            2578   12936#
EMT66   070422            2585   12936#
EMT67   070434            2592   12936#
EMT7    067570            2252   12936#
EMT70   070446            2599   12936#
EMT71   070456            2606   12936#
EMT72   070466            2613   12936#
EMT73   070504            2621   12936#
EMT74   070522            2628   12936#
EMT75   070532            2635   12936#
EMT76   070552            2642   12936#
EMT77   070562            2649   12936#
```

```
ENRGDT  067526          12936#
EQUALS  066137          12936#
ERR   = 040000           1706#  4584   6213   6322   6437   6552   6676   9918   9920   9936   9970
ERRNMB  043002           9109#  9110#  9113   9125   9130   9222#
ERRTYP  042244           9088#  12465
ERRVEC= 000004           1624#  3945   3946#  3957#  4192   4193   4194#  4195#  4206#  4207#  4211#  4212#  4238
                         4239   4240#  4241#  4245#  4246#  4257#  4258#  6938   6939   6940#  6941#  6997#  6998#
                         8102   8103   8104#  8105#  8183#  8184#  9490   9491   9495#  9496#  9527#  9528#  9563
                         9564   9568#  9569#  9594#  9595#  9606   9607   9608#  9609#  9615#  9623#  9624#  9638
                         9639   9640#  9641#  9665#  9666#  10513  10514  10517#  10518#  10551#  10552#  10799  10800
                        10801#  10802#  10813#  10814#  12325  12326#  12328#  12331#
ERTY00  043010           9095   9227#
ERTY01  043016           9103   9228#
ERTY02  043026           9112   9230#
ERTY03  043035           9118   9232#
ERTY04  043043           9214   9234#
ESRC  = 004000           1780#
FER   = 000020           1730#  1736   10250
FIND  = 000001           4495#  4496   4503#  4504   4657#  4658   4664#  4665   4869#  4870   4876#  4877   4884#
                         4885   4906#  4908#  4929#  4930   4936#  4937   4943#  4944   4965#  4967#  5005#  5006
                         5012#  5013   5019#  5020   5040#  5042#  5081#  5082   5091#  5092   5112#  5114#  5140#
                         5141   5147#  5148   5154#  5155   5176#  5178#  5201#  5203#  5205#  5206   5211#  5212
                         5218#  5219   5225#  5226   5242#  5244#  5273#  5274   5280#  5281   5287#  5288   5305#
                         5307#  5327#  5328   5334#  5337#  5343#  5344   5350#  5351   5368#  5370#  5391#  5392
                         5410#  5413#  5419#  5420   5426#  5427   5444#  5446#  5472#  5475#  5496#  5497   5509#
                         5512#  5518#  5519   5525#  5526   5543#  5545#  5588#  5589   5601#  5604#  5621#  5623#
                         5679#  5680   5686#  5687   5704#  5706#  5759#  5760   5778#  5779   5785#  5788#  5794#
                         5795   5801#  5802   5819#  5821#  5892#  5894#  5915#  5916   5934#  5937#  5943#  5944
                         5950#  5951   5969#  5971#  5999#  6000   6021#  6024#  6030#  6031   6037#  6038   6055#
                         6057#  6077#  6078   6124#  6126#  6198#  6199   6239#  6241#  6307#  6308   6348#  6350#
                         6422#  6423   6463#  6465#  6537#  6538   6578#  6580#  6661#  6662   6702#  6704#  6709#
                         6712#  6713   6742#  6743   6748#  6749#  6750   6755#  6756   6778#  6780#  6785#  6814#
                         6815   6848#  6849   6872#  6874#  6881#  6882   6897#  6898   6903#  6904   6930#  6932#
                         6952#  6953   6964#  6965   7015#  7017#  7043#  7044   7050#  7051   7057#  7060#  7066#
                         7067   7088#  7090#  7116#  7117   7123#  7124   7130#  7133#  7139#  7140   7162#  7164#
                         7188#  7189   7195#  7196   7202#  7205#  7211#  7212   7241#  7243#  7271#  7272   7278#
                         7279   7285#  7288#  7294#  7295   7321#  7323#  7351#  7352   7358#  7359   7365#  7368#
                         7374#  7375   7402#  7404#  7431#  7432   7438#  7439   7445#  7448#  7454#  7455   7482#
                         7484#  7511#  7512   7518#  7519   7525#  7528#  7534#  7535   7558#  7560#  7588#  7589
                         7601#  7602   7607#  7608   7614#  7615   7622#  7625#  7631#  7632   7653#  7655#  7683#
                         7684   7702#  7703   7709#  7710   7717#  7720#  7726#  7727   7752#  7754#  7782#  7783
                         7801#  7802   7808#  7809   7816#  7819#  7825#  7826   7889#  7890   7903#  7906#  7912#
                         7913   7930#  7932#  7971#  7972   7978#  7979   7985#  7988#  7994#  7995   8016#  8018#
                         8048#  8056#  8057   8063#  8064   8070#  8073#  8079#  8080   8131#  8132   8196#  8198#
                         8226#  8227   8233#  8234   8253#  8254   8260#  8261   8267#  8270#  8276#  8277   8305#
                         8307#  8337#  8338   8344#  8345   8364#  8365   8371#  8372   8378#  8381#  8387#  8388
                         8419#  8421#  8449#  8450   8462#  8463   8469#  8472#  8478#  8479   8514#  8516#  8545#
                         8546   8559#  8562#  8568#  8569   8602#  8604#  8632#  8633   8645#  8646   8652#  8655#
                         8661#  8662   8696#  8698#  8736#  8737   8750#  8753#  8759#  8760   8776#  8778#  8817#
                         8818   8824#  8825   8831#  8834#  8840#  8841   8857#  8859#  8896#  8897   8903#  8904
                         8910#  8913#  8919#  8920   8958#  8960#  8986#  8987   8993#  8994   9000#  9003#  9009#
                         9010
FMT16 = 010000           1805#  5984   6006   6177   6286   6401   6516   6640   6733   7734   7736   7833   7835
                         8107   8159   8176   8285   8287   8399   8404   8576   8578   8669   8671   8865   10610
                        11601
FNCDTB  066770           6204   6212   6313   6321   6428   6436   6543   6551   6667   6675   9964   12936#
```

DZRMCA.P11 29-JUL-77 13:33 CROSS REFERENCE TABLE -- USER SYMBOLS

```
FNCMSK= 000077          1648#  11397
F0    = 000002          1646#   9863
F1    = 000004          1645#   9863
F2    = 000010          1644#   9863  10437
F3    = 000020          1643#   9863
F4    = 000040          1642#   9863
GET     044046          4285    4357   4489   4539   4576   4651   4637   4748   4805   4863   4923   4990   4999
                        5075    5134   5205   5267   5321   5385   5460   5490   5559   5582   5637   5667   5753
                        5772    5874   5909   5993   6071   6100   6192   6301   6416   6531   6655   6749   6814
                        6848    6903   6958   7037   7110   7182   7265   7345   7425   7505   7582   7601   7677
                        7696    7776   7795   7883   7953   7965   8050   8125   8220   8247   8331   8358   8443
                        8539    8626   8730   8799   8811   8890   8980   9306   9330   9357   9380   9406   9481#
GETBUF  001326          2112#   9493
GETINX  001504          2183#   4282*  4283*  4536*  4537*  4538*  9328*  9329*  9378*  9379*  9439   9494
GETSTS  043762          4356    4488   4575   4649   4696   4861   4921   4989   5073   5132   5201   5265   5320
                        5383    5459   5558   5636   5745   5843   5907   5991   6070   6130   6245   6354   6469
                        6584    6709   6785   6937   7035   7108   7180   7263   7343   7423   7503   7574   7669
                        7768    7876   7946   8036   8117   8213   8324   8435   8531   8618   8723   8792   8878
                        8978    9305   9356   9399   9435#
GMS     = ****** U      1955    3983   9047   9054   12820  12821  12822  12823  12824  12825  12827  12829  12830
                       12831   12832  12833  12834
GO      = 000001        1647#   4643   4798   4855   4915   4983   5067   5121   5185   5194   5250   5314   5377
                        5453    5483   5552   5630   5660   5739   5846   5901   5985   6064   6164   6171   6280
                        6388    6395   6503   6510   6618   6634   6727   6807   6832   6841   7024   7097   7169
                        7251    7330   7411   7491   7564   7659   7758   7866   7939   8025   8039   8110   8206
                        8239    8316   8351   8428   8524   8611   8715   8783   8866   8882   8967   9340   9389
                        9651    9762   9774   9779   9795   9885   9888
GTSWR = 104407          3978    4167   12827#
HCE   = 000200          1727#   1736   10213  10218  10233  10235  10248  10263  12936
HCI   = 002000          1807#   5984   6006   10211
HCRC  = 000400          1726#   1736   10220
HELP    105642          4031   12936#
HELPOS  066147          4025   12936#
HT    = 000011          1534#   9151   12262  12303
IAE   = 002000          1724#   6429   6431   7623   7718   7817   8455   8457   8459   8470   8551   8553   8555
                        8560    8638   8640   8642   8653   10117  10414  10429  10455  10601  10618  11117  11171
                       11174   11592  11610  12936
IDXMSK= 000077          1883#   9507   9510   9580   9583
IE    = 000100          1897#   1930#  5739   5846
ILF   = 000001          1734#   6205   6207   9989   9991   11011  11071  11075  11117  11138  11141  12936
ILF02 = 000002          1653#
ILF24 = 000024          1663#
ILF26 = 000026          1664#
ILF30 = 000030          1666#
ILF32 = 000032          1667#
ILF34 = 000034          1668#
ILF36 = 000036          1669#
ILF40 = 000040          1670#
ILF42 = 000042          1671#
ILF44 = 000044          1672#
ILF46 = 000046          1673#
ILF54 = 000054          1676#
ILF56 = 000056          1677#
ILF64 = 000064          1680#
ILF66 = 000066          1681#
ILF74 = 000074          1684#
```

```
ILF76 = 000076          1685#   6224    6333    6448    6563    6687    6764
ILR   = 000002          1733#   6972    6985    6999   11011   11057   11061   11325   11329   11332
ILRG50= 000050          1869#
ILRG52= 000052          1870#
ILRG54= 000054          1871#
ILRG56= 000056          1872#
ILRG60= 000060          1873#
ILRG62= 000062          1874#
ILRG64= 000064          1875#
ILRG66= 000066          1876#
ILRG70= 000070          1877#
ILRG72= 000072          1878#
ILRG74= 000074          1879#
ILRG76= 000076          1880#
IOTVEC= 000020          1629#   3929*   3930*
IPCK0 = 000001          1934#
IPCK1 = 000002          1933#
IPCK2 = 000004          1932#
IPCK3 = 000010          1931#
IR    = 000100          1911#   4378   10874
IVC   = 010000          1840#   5087    6314    6316    7895    7897    7899    7905    8742    8744    8746    8752   10032
                       10033   10636   10759   11185   11192   11195   11264   11627   11651   11654   11741   11835   12936
LBC   = 002000          1842#   4810    4813    4814
LBT   = 002000          1710#   4584   10192   10416   10418
LF    = 000012          1535#   9149   12006   12297   12303   12936
LS    = 000004          1789#   4401   10898   11450
LSC   = 004000          1841#   5253
LST   = 000002          1790#   4401   10898   11450
MCLK  = 004000          1762#
MCPE  = 020000          1891#   4546    9762    9779    9792    9795   11572   11577
MDF   = 000100          1767#   4741
MDPE  = 000400          1908#   4551   10288
MEDEMB  001472          2171#   4178*   9339*
MFGFIL  103622         12936#
MI    = 000004          1770#
MIXED   067100         12936#
MOC   = 000400          1765#   4741    6153    6268    6377    6492    6607    6627    6798    6823
MOH   = 020000          1797#
MOL   = 010000          1708#   4581    4585    4612    4708    4711    4759    4762    4763   10697   10707   10708   10713
                       10718   10746   10997   11000   11161   11238   11239   11246   11251   11410   11691   11702   11703
                       11707   11710   11812   11813   11817   11822
MRD   = 002000          1763#
MS    = 000040          1768#
MSC   = 000002          1771#
MSE   = 100000          1848#
MSER  = 000200          1766#   4741
MUR   = 001000          1764#   4741    6153    6268    6377    6492    6607    6798    6823
MWD   = 000010          1788#   4402   10899   11451
MWP   = 000010          1769#   4741    6492
MXF   = 001000          1907#   4551   10177
NOTMSK= 115760          1736#   5335    5411    5510    5786    5935    6022    7058    7131    7203    7286    7366    7446
                        7526    7623    7718    7817    7904    7986    8071    8268    8379    8470    8560    8653    8751
                        8832    8911    9001
NED   = 010000          1904#   4247    4551    9744    9748   10526
NEM   = 004000          1905#  10162
NOP   = 000000          1652#   4502    5314    5483    5660    5858    6131    6246    6355    6470    6585    6710    6832
```

```
NSA    = 100000          1795#
OCC    = 100000          1776#
OFD    = 000200          1808#  5984   6006
OFFSET= 000014           1658#  5185   5377   5453   5552   5630   5739   5846   6064   6841   8025   8866
OM     = 000001          1715#  4612   5397   5400   5401   5921   5925   6083   6086   6087   6105   6108  11238
                        11294  11297  11409
ONES     067140         12936#
OPE    = 020000          1839#
OPI    = 020000          1721#  6668   6670  10016  10691  10715  11011  11029  11033  11117  11154  11157  11248
                        11684  11687  11712  11819  12936
OR     = 000300          1910#
PACACK= 000022           1662#
PAKACK= 000022           1661#  1662   4643   4855   6164   6388   6503   6618   6807   9340
PAR    = 000010          1731#  6909   6911   9807   9811   9823  10583  11117  11122  11127  11552  11559
PAT    = 000020          1913#  6889
PDA    = 000400          1783#
PGE    = 002000          1906#  4551
PGM    = 001000          1711#  4612  11409
PHA    = 000200          1784#
PIP    = 020000          1707#  4584   4714   4717   4718   4765   4768   4995   7958   7961   8804   8808   9387
                        10707  10728  10733  11238  11309  11314  11409  11702  11723  11728  11812  11849  11854
PIRQ   = 177772          1541#
PIRQVE= 000240           1635#
PLFS   = 002000          1781#
PRIERR   045042          4657   4869   4929   5005   5081   5140   5211   5273   5327   5391   5496   5588   5759
                         5778   5915   5999   6077   6198   6307   6422   6537   6661   6755   6964   7043   7116
                         7188   7271   7351   7431   7511   7607   7702   7801   7889   7971   8056   8131   8226
                         8253   8337   8364   8449   8545   8632   8736   8817   8896   8986   9710#
PROMPT   066141          4104   4112  12936#
PR0    = 000000          1558#
PR1    = 000040          1559#
PR2    = 000100          1560#
PR3    = 000140          1561#  11985
PR4    = 000200          1562#
PR5    = 000240          1563#
PR6    = 000300          1564#  3916   4173   4195   4241   5716   5725   5830   5833   6941   8105   9496   9569
                         9609   9641  10518  10802  11991
PR7    = 000340          1565#  5723   5839
PS     = 177776          1538#  1539
PSEL   = 002000          1893#
PSW    = 177776          1539#
PUT      044316          4326   4348   4455   4476   4531   4644   4691   4742   4799   4838   4847   4856   4916
                         4975   4984   5050   5059   5068   5127   5186   5195   5260   5315   5378   5454   5484
                         5553   5576   5631   5652   5661   5732   5740   5847   5859   5902   5986   6065   6094
                         6145   6154   6165   6186   6260   6269   6295   6369   6378   6389   6410   6484   6493
                         6504   6525   6599   6608   6619   6628   6649   6721   6742   6790   6799   6808   6824
                         6833   6842   6897   7030   7103   7175   7258   7337   7418   7498   7595   7690   7789
                         7860   7877   7947   8031   8042   8119   8214   8240   8325   8352   8437   8533   8620
                         8706   8724   8793   8873   8883   8973   9343   9392   9557#
PUTBUF   001376          2139#  9566
PUTINX   001533          2188#  4323*  4324*  4340   4452*  4453*  4468   4528*  4529*  4641*  4642*  4688*  4689*
                         4739*  4740*  4796*  4797*  4835*  4836*  4844*  4845*  4853*  4854*  4913*  4914*  4972*
                         4973*  4981*  4982*  5047*  5048*  5056*  5057*  5065*  5066*  5122   5183*  5184*  5192*
                         5193*  5254   5312*  5313*  5375*  5376*  5451*  5452*  5481*  5482*  5550*  5551*  5574*
                         5575*  5628*  5629*  5649*  5650*  5658*  5659*  5730*  5731*  5737*  5738*  5844*  5845*
                         5856*  5857*  5899*  5900*  5976   6062*  6063*  6092*  6093*  6142*  6143*  6151*  6152*
```

```
                    6162*  6163*  6178   6257*  6258*  6266*  6267*  6287   6366*  6367*  6375*  6376*  6386*
                    6387*  6402   6481*  6482*  6490*  6491*  6501*  6502*  6517   6596*  6597*  6605*  6606*
                    6616*  6617*  6625*  6626*  6641   6718*  6719*  6734   6787*  6788*  6796*  6797*  6805*
                    6806*  6821*  6822*  6830*  6831*  6839*  6840*  6892   7025   7098   7170   7252   7331
                    7412   7492   7568   7663   7762   7857*  7858*  7870   7940   8026   8040*  8041*  8111
                    8207   8318   8429   8525   8612   8703*  8704*  8716   8786   8867   8880*  8881*  8968
                    9341*  9342*  9390*  9391*  9567   9814
PWRVEC= 000024       1630*  3935*  3936*  12839* 12840* 12849* 12855* 12867* 12868*
QSTMRK  066145       4020   4122   12936#
RCLSTS  055002       4936   5147   5218   9413   11108#
RD    = 000070       1682#
RDCHR = 104411       4012   4026   4036   4105   4113   12692  12830#
RDLIN = 104412      12762  12831#
RDOCT = 104413       4048   4064   4085   12832#
RDY   = 000200       1896#  4365   9651   9652   9758   9761   10849
READY   006752       4017   4171#  9022   9070
RECAL = 000006       1655#  4915   4983   5067   5121   5194   9389
RESREG= 104415       9218   12834#
RESVEC= 000010       1625#
REX   = 010000       1779#
RG    = 040000       1777#
RGDTPT  067100      12936#
RH    = 000072       1683#
RIP   = 000020       1660#  5985
RLEASE= 000012       1657#
RMAS  = 000016       1857#  5574   5730
RMASI   001344       2122#  4390   11430  12936
RMASO   001414       2149#  5573*  5729*
RMBA  = 000004       1939#  4200*  4201   4341   4469   6179   6288   6403   6518   6642   6735
RMBAE = 000050       1942#  6978   6981
RMBAI   001332       2117#  4371   10364  10366  10860  12936
RMBAO   001402       2144#  4333*  4462*  6173*  6282*  6397*  6512*  6636*  6729*  10360  10363
RMCC  = 000036       1864#
RMCCI   001364       2130#
RMCCO   001434       2157#
RMCS1 = 000000       1853#  1937#  4197*  4243   4345   4473   4528   4536   4641   4796   4853   4913   4981
                     5065   5125   5183   5192   5258   5312   5375   5451   5481   5550   5628   5658   5737
                     5844   5856   5899   5980   6062   6162   6184   6293   6386   6408   6501   6523   6616
                     6647   6740   6805   6830   6839   6980*  7028   7101   7173   7255   7334   7415   7495
                     7572   7667   7766   7874   7944   8029   8040   8115   8211   8322   8433   8529   8616
                     8721   8790   8871   8880   8971   9341   9390   9498   9571   9650   10524
RMCS1I  001326       2115#  4363   4544   9737   9739   9741   9758   9760   9763   9774   9778   9780   9792
                     9794   9796   9885   9887   9898   9941   10347  10847  11396  11572  11576  11578  12936
RMCS10  001376       2142#  4337*  4466*  4502*  4530*  4643*  4798*  4855*  4915*  4983*  5067*  5121*  5185*
                     5194*  5250*  5314*  5377*  5453*  5483*  5552*  5630*  5660*  5739*  5846*  5858*  5901*
                     5985*  6064*  6164*  6171*  6172*  6280*  6281*  6388*  6395*  6396*  6503*  6510*  6511*
                     6618*  6634*  6635*  6727*  6728*  6807*  6832*  6841*  7024*  7097*  7169*  7251*  7330*
                     7411*  7491*  7564*  7659*  7758*  7866*  7939*  8025*  8039*  8110*  8206*  8239*  8316*
                     8351*  8428*  8524*  8611*  8715*  8783*  8866*  8882*  8967*  9340*  9389*  9862   10378
RMCS2 = 000010       1940#  4202*  4203   4244   4346   4355*  4474   4537   6893   9497   9570   10523*  10525
                    10804*  10806*
RMCS2I  001336       2119#  4375   4550   9719   9744   9746   9747   9933   10004  10105  10148  10163  10178
                    10289   10361  10869  12936
RMCS20  001406       2146#  4338*  4467*  6890*
RMCS3 = 000052       1943#  6987
RMDA  = 000006       1854#  5124   5255   5977   6182   6291   6406   6521   6645   6738   6894   7027   7100
```

```
                    7172   7253   7332   7414   7493   7570   7666   7763   7872   7941   8028   8112   8208
                    8321   8432   8526   8615   8719   8788   8868   8970  11603
RMDAI   001334      2118#  6010   6012  10460  12936
RMDAO   001404      2145#  5120*  5249*  5982*  6175*  6284*  6399*  6514*  6638*  6731*  6891*  7022*  7095*
                    7168*  7250*  7329*  7410*  7490*  7567*  7636*  7637   7661*  7731*  7732   7737*  7759*
                    7868*  7937*  8023*  8108*  8203*  8282*  8283   8289*  8311*  8395*  8396   8405*  8426*
                    8483*  8484   8522*  8573*  8574   8579*  8609*  8713*  8785*  8863*  8965*  10386  10387
                   10606  10608  10612  11597  11599
RMDB  = 000022       1941#  4204*  4205   4323   4452   9446
RMDBI   001350       2124#
RMDBO   001420       2151#  4322*  4451*
RMDC  = 000034       1863#  5123   5978   6092   6183   6292   6407   6522   6646   6739   7026   7099   7171
                    7254   7333   7413   7494   7571   7665   7764   7873   7942   8027   8113   8209   8320
                    8430   8527   8614   8718   8787   8869   8969
RMDCI   001362       2129#  6015   6017  10472  12936
RMDCO   001432       2156#  5119*  5983*  6091*  6176*  6285*  6400*  6515*  6639*  6732*  7023*  7096*  7160*
                    7216   7218*  7249*  7328*  7381   7383*  7409*  7461   7463*  7489*  7541   7543*  7566*
                    7660*  7760*  7830*  7831   7836*  7869*  7938*  8024*  8109*  8204*  8317*  8350*  8427*
                    8523*  8610*  8666*  8667   8672*  8712*  8784*  8864*  8966*  10388  10603  11594
RMDS  = 000012       1855#  9328   9378
RMDSI   001340       2120#  4424   4581   4583   4586   4609   4611   4613   4708   4710   4714   4716   4720
                    4722   4759   4761   4765   4767   4771   4773   4789   4791   4995   5397   5399   5403
                    5405   5465   5468   5502   5504   5564   5566   5594   5596   5642   5644   5672   5675
                    5921   5924   5927   5929   6083   6085   6105   6107   6213   6217   6322   6326   6437
                    6441   6552   6556   6676   6680   7958   7960   8804   8807   9337   9387   9776   9872
                    9874   9919   9936   9970   9974  10030  10057  10192  10417  10642  10697  10706  10713
                   10717  10719  10728  10732  10734  10743  10745  10747  10757  10761  10763  10942  10983
                   10985  10987  10997  10999  11001  11161  11199  11237  11246  11250  11252  11262  11266
                   11267  11278  11280  11282  11294  11296  11298  11309  11313  11315  11408  11658  11691
                   11701  11707  11709  11711  11723  11727  11729  11739  11743  11745  11755  11757  11759
                   11811  11817  11821  11823  11833  11837  11838* 11839  11849  11853  11855  12936
RMDSO   001410       2147#
RMDT  = 000026       1860#  4282
RMDTI   001354       2126#  4284*  4289   4291   12936
RMDTO   001424       2153#
RMEC1 = 000044       1867#
RMEC1I  001372       2133#  12936
RMEC1O  001442       2160#
RMEC2 = 000046       1868#  9441
RMEC2I  001374       2134#  4408   9440  10910  11474  12936
RMEC2O  001444       2161#
RMER1 = 000014       1856#  4342   4470   4972   5257   5649   6718   7943   8789   9815
RMER1I  001342       2121#  4385   4598   4600   4604   4606   6206   6430   6545   6669   6855   6858   6909
                    6912   6971   8455   8458   8551   8554   8638   8641   9807   9822   9824   9914   9990
                   10017  10060  10075  10118  10195  10221  10236  10251  10304  10323  10326  10414  10429
                   10444  10455  10583  10590  10617  10691  10694  10715  10887  11011  11015  11017  11018
                   11029  11031  11032  11043  11045  11046  11057  11059  11060  11071  11073  11074  11117
                   11122  11126  11128  11138  11140  11142  11154  11156  11158  11171  11173  11175  11248
                   11325  11329  11331  11333  11343  11345  11347  11357  11361  11363  11421  11552  11558
                   11560  11609  11684  11686  11688  11712  11819  11930  12936
RMER1O  001412       2148#  4334*  4463*  4974*  5251*  5651*  6720*  7936*  8782*  9811
RMER2 = 000042       1866#  4343   4471   5256
RMER2I  001370       2132#  4420   4590   4592   4596   4617   4619   4726   4728   4732   4734   4777   4779
                    4783   4785   4810   4812   5087   6315   7895   7898   8742   8745   9809   9916  10034
                   10090  10266  10585  10636  10639  10650  10678  10681  10730  10759  10932  11124  11185
                   11192  11194  11196  11209  11211  11213  11223  11225  11227  11264  11311  11359  11483
```

```
                    11554   11627   11631   11651   11653   11655   11668   11670   11672   11725   11741   11835   11851
                    11865   11871   11873   11875   11885   11887   11889   11933   12936
RMER20  001440       2159#   4335#   4464#   5252#   5253#
RMLA  = 000020       1858#   8140    8141    8150
RMLAI   001346       2123#  12936
RMLAO   001416       2150#
RMMR1 = 000024       1859#   4344    4472    4688    4739    4835    4844    5047    5056    6142    6151    6257    6266
                     6366    6375    6481    6490    6596    6605    6625    6787    6796    6821    7857    7871    8703
                     8717
RMMR1I  001352       2125#   4400    4702    4705    4753    4756   10897   11449   12936
RMMR10  001422       2152#   4336#   4465#   4690#   4741#   4837#   4846#   5049#   5058#   6144#   6153#   6259#   6268#
                     6368#   6377#   6483#   6492#   6598#   6607#   6627#   6789#   6798#   6823#   7859#   7867#   8705#
                     8711#
RMMR2 = 000040       1865#
RMMR2I  001366       2131#   4412   10919   11461   12936
RMMR20  001436       2158#
RMOF  = 000032       1862#   5979    6181    6290    6405    6520    6644    6737    7569    7664    7765    8114    8210
                     8319    8431    8528    8613    8720    8870
RMOFI   001360       2128#   6005   10211   10321   11601   12936
RMOFO   001430       2155#   5984#   6177#   6286#   6401#   6516#   6640#   6733#   7565#   7662#   7734    7736#   7761#
                     7833    7835#   8107#   8159    8176    8178#   8205#   8285    8287#   8312#   8399    8404#   8425#
                     8521#   8576    8578#   8608#   8669    8671#   8714#   8865#  10610   12936
RMR   = 000004       1732#   6055    6857   11011   11043   11047   11325   11343   11346
RMSN  = 000030       1861#
RMSNI   001356       2127#  12936
RMSNO   001426       2154#
RMWC  = 000002       1938#   4198#   4199    6180    6289    6404    6519    6643    6736
RMWCI   001330       2116#  10345   10357   10375   10439   12936
RMWCO   001400       2143#   6174#   6283#   6398#   6513#   6637#   6730#  10358   10376
RQA   = 100000       1817#   4413   10920   11462
RQB   = 040000       1818#   4413   10920   11462
RTC   = 000016       1659#   5901
SAOMSK= 000037       1701#
SAVREG= 104414       9089   12833#
SA1   = 000001       1696#
SA16  = 000020       1692#
SA2   = 000002       1695#
SA4   = 000004       1694#
SA8   = 000010       1693#
SC    = 100000       1889#   4364    9762    9779    9795   10848
SCHSTS  057346       8260    8371    8903   11543#
SCTMSG  066026      12936#
SCTMSK= 003700       1752#
SCO   = 000100       1750#
SC1   = 000200       1749#
SC2   = 000400       1748#
SC3   = 001000       1747#
SC4   = 002000       1746#
SEARCH= 000030       1665#   8110    8239    8351    8428    8524    8611    8715    8783    8882    9938   10339
SECERR  045674       4503    4664    4884    4943    5019    5091    5154    5225    5287    5350    5426    5525    5686
                     5801    5950    6037    7066    7139    7211    7294    7374    7454    7534    7631    7726    7825
                     7912    7994    8079    8276    8387    8478    8568    8661    8759    8840    8919    9009    9858#
SEEK  = 000004       1654#   7024    7097    7169    7251    7330    7411    7491    7564    7659    7758    7866    7939
                     8039    8206    8316    8967
SEKSTS  051750       7050    7123    7195    7278    7358    7438    7518    7614    7709    7808    8063    8233    8344
                     8993   10572#
```

```
SHUT    061660          11999#  12533   12616
SIZCLK  044534           4154    9605#
SKI   = 040000           1838#   4421    4617    4726    4729    4777    4780    4781   10602   10605   10621   10651   10653
                        10657   10668   10730   10933   11185   11209   11212   11311   11593   11596   11613   11627   11632
                        11634   11639   11725   11851   11866   11885   11888
SNGPRT= 020024           1800#   4289    4293
STACK = 001100           1529#   3927    4187    4227    4271    4310    4440    4517    4564    4630    4677    4824    4902
                         4960    5036    5108    5171    5238    5301    5364    5439    5538    5616    5699    5814    5888
                         5964    6050    6119    6234    6343    6458    6573    6697    6773    6867    6925    7010    7083
                         7156    7236    7316    7397    7477    7552    7648    7747    7846    7925    8011    8091    8192
                         8301    8414    8510    8597    8691    8771    8853    8941
STANDA  006002           3991    4002#
START   005322           1975    3911#   4216   12005
STCORV  060712           5012    5343    5419    5518    5679    5794    5943    6030    7978    8462    8645    8824   11804#
STKLMT= 177774           1540#
STOP    061630          11982#  12319
STSD1   103512          12936#
STSD2   103530          12936#
STSD3   103550          12936#
STSD4   103562          12936#
STSF    103613          12936#
STSH1   103155          12936#
STSH2   103232          12936#
STSH3   103321          12936#
STSH4   103357          12936#
SWR     001154           2028#   3925    3947*   3949    3955*   3962*   3976    4165    9090   12320   12334   12337   12352
                        12359   12456   12463   12475   12479   12536   12575   12630*  12847   12860*
SWREG   000176           1957#   3955    3976    4165   12536   12575   12598
SW0   = 000001           1593#
SW00  = 000001           1583#   1593
SW01  = 000002           1582#   1592
SW02  = 000004           1581#   1591
SW03  = 000010           1580#   1590
SW04  = 000020           1579#   1589
SW05  = 000040           1578#   1588
SW06  = 000100           1577#   1587
SW07  = 000200           1576#   1586
SW08  = 000400           1575#   1585
SW09  = 001000           1574#   1584
SW1   = 000002           1592#
SW10  = 002000           1573#
SW11  = 004000           1572#
SW12  = 010000           1571#
SW13  = 020000           1570#   9090
SW14  = 040000           1569#
SW15  = 100000           1568#
SW2   = 000004           1591#
SW3   = 000010           1590#
SW4   = 000020           1589#
SW5   = 000040           1588#
SW6   = 000100           1587#
SW7   = 000200           1586#
SW8   = 000400           1585#
SW9   = 001000           1584#
TAOMSK= 003400           1700#
TAG   = 020000           1819#
```

```
TAGADR= 001114          2000#   2007
TAP   = 040000          1796#
TA1   = 000400          1691#
TA2   = 001000          1690#
TA4   = 002000          1689#
TBITVE= 000014          1626#
TIMOUT  044656          4650    4862    4922    4998    5074    5133    5203    5266    5384    5744    5851    5908    5992
                        6191    6300    6415    6530    6654    6748    7036    7109    7181    7264    7344    7424    7504
                        7700    7695    7794    7964    8037    8048    8124    8219    8246    8330    8357    8442    8538
                        8625    8729    8810    8879    8889    8979    9400    9634#
TKVEC = 000060          1633#  12513*  12514*
TPVEC = 000064          1634#
TRAPVE= 000034          1632#   3933*   3934*
TRE   = 040000          1890#   4530    4546    9762    9779    9795    9940    9942   10347
TRTVEC= 000014          1627#
TST   = 010000          1820#   4414   10921   11463
TSTNME  043000          9101*   9102*   9104    9221#
TSTPRP  043046          4906    4965    5040    5112    5176    5242    5305    5368    5444    5543    5621    5704    5819
                        5892    5969    6055    6124    6239    6348    6463    6578    6702    6778    6872    6930    7015
                        7088    7162    7241    7321    7402    7482    7558    7653    7752    7930    8016    8196    8305
                        8419    8514    8602    8696    8776    8857    8958    9269#
TSTQUE  001446          2167#   4137    4138*   4177    4189    4229    4273    4312    4442    4519    4566    4632    4679
                        4826    4904    4962    5038    5110    5173    5240    5303    5366    5441    5540    5618    5701
                        5816    5890    5966    6052    6121    6236    6345    6460    6575    6699    6775    6869    6927
                        7012    7085    7158    7238    7318    7399    7479    7554    7650    7749    7848    7927    8013
                        8093    8194    8303    8416    8512    8599    8693    8773    8855    8943    9017    9019*   9023*
                       10520   10805   10872   11433
TST1    007004          4182#  12379
TST10   011502          4627#  12386
TST11   011662          4674#  12387
TST12   012734          4821#  12388
TST13   013234          4895#  12389
TST14   013466          4953#  12390
TST15   014010          5029#  12391
TST16   014334          5101#  12392
TST17   014612          5164#  12393
TST2    007202          4224#  12380
TST20   015104          5235#  12394
TST21   015354          5298#  12395
TST22   015604          5361#  12396
TST23   016130          5436#  12397
TST24   016550          5535#  12398
TST25   017114          5613#  12399
TST26   017506          5696#  12400
TST27   020162          5811#  12401
TST3    007366          4268#  12381
TST30   020514          5885#  12402
TST31   021036          5961#  12403
TST32   021420          6047#  12404
TST33   021736          6116#  12405
TST34   022522          6231#  12406
TST35   023246          6340#  12407
TST36   024036          6455#  12408
TST37   024622          6570#  12409
TST4    007530          4303#  12382
TST40   025446          6694#  12410
```

```
TST41   026004          6770#  12411
TST42   026444          6864#  12412
TST43   026670          6922#  12413
TST44   027232          7003#  12414
TST45   027534          7076#  12415
TST46   030036          7149#  12416
TST47   030364          7229#  12417
TST5    010414          4437#  12383
TST50   030706          7309#  12418
TST51   031234          7390#  12419
TST52   031562          7470#  12420
TST53   032110          7549#  12421
TST54   032476          7645#  12422
TST55   033112          7744#  12423
TST56   033524          7843#  12424
TST57   034064          7922#  12425
TST6    010752          4514#  12384
TST60   034426          8004#  12426
TST61   034774          8088#  12427
TST62   035446          8189#  12428
TST63   036124          8298#  12429
TST64   036632          8411#  12430
TST65   037212          8507#  12431
TST66   037600          8594#  12432
TST67   040204          8688#  12433
TST7    011156          4561#  12385
TST70   040564          8768#  12434
TST71   041130          8850#  12435
TST72   041476          8934#  12436
TYPBN = 104406          9211   12825#
TYPDS = 104405          9051   9058   9205   12824#
TYPE  = 104401          3970   4011   4014   4020   4025   4028   4031   4035   4038   4044   4047   4053   4058
                        4063   4069   4079   4084   4090   4103   4104   4107   4112   4115   4122   9045   9052
                        9059   9094   9095   9103   9112   9118   9127   9165   9171   9175   9188   9194   9196
                        9214   9216   12000  12072  12138  12207  12267  12458  12466  12530  12542  12596  12597
                       12600  12611  12621  12632  12651  12699  12705  12710  12714  12719  12720  12722  12725
                       12729  12820#  12869
TYPOC = 104402          4046   9208   12599  12821#
TYPON = 104404         12823#
TYPOS = 104403          4060   4081   9098   9106   9115   9121   12822#
UBUSQS  066203          4035   12936#
UNS   = 040000          1720#  4386   4598   4601   4604   4974   5651   6720   7936   7986   8782   8832   10888
                       11011  11015  11019  11325  11357  11362
UNTMSK= 000007          1921#  6888   9720   9722
UPE   = 020000          1903#  4551   10104
USE   = 040000          1849#
USRFIL  104632         12936#
U0    = 000001          1918#
U1    = 000002          1917#
U2    = 000004          1916#
VV    = 000100          1714#  4612   4789   4792   9337   10030  10642  10707  10708  10757  10762  10983  10986
                       11199  11238  11239  11262  11268  11409  11658  11702  11703  11739  11744  11812  11813
                       11833  11838
WC    = 000040          1786#  4401   10838  11450
WCD   = 000050          1674#
WCE   = 040000          1902#  4551   10003  12936
```

DZRMCA.P11 29-JUL-77 13:33 CROSS REFERENCE TABLE -- USER SYMBOLS

```
WCEHI = 010000          1927#
WCELO = 004000          1928#
WCF   = 000040          1729#   1736   10074
WCH   = 000052          1675#
WD    = 000060          1678#
WH    = 000062          1679#
WLE   = 004000          1723#   1736    6544    6546   10055   10059   10072   10087   10102   12936
WRL   = 004000          1709#   4609    4720    4723    4771    4774    4775   10057
ZEROS   067202         12936#
$APTHD  001100          1987    1993#
$ASTAT= ****** U       12912   12927
$ATYC   065604         12883   12885#
$ATY1   065560         12881#
$ATY3   065566         12252   12882#
$ATY4   065576         12471   12884#
$AUTOB  001150          2025#   3980*   4169*   12559   12746
$BASE   001276          2094#   4045    4055*   4188    4228    4272    4311    4441    4518    4565    4631    4678    4825
                        4903    4961    5037    5109    5172    5239    5302    5365    5440    5539    5617    5700    5815
                        5889    5965    6051    6120    6235    6344    6459    6574    6698    6774    6868    6926    7011
                        7084    7157    7237    7317    7398    7478    7553    7649    7748    7847    7926    8012    8092
                        8193    8302    8415    8511    8598    8692    8772    8854    8942    9492    9565    9642   10519
                       10803   12936
$BDADR  001136          2020#   12936
$BDDAT  001142          2022#   4363*   4364*   4366    4371*   4375*   4380    4385*   4386*   4390*   4396*   4400*   4401*
                        4403    4408*   4412*   4413*   4415    4420*   4421*   4424*   4425*   4427    4544*   4546*   4550*
                        4551*   4586*   4592*   4600*   4606*   4613*   4619*   4705*   4710*   4711*   4716*   4717*   4722*
                        4723*   4728*   4729*   4734*   4735*   4756*   4761*   4762*   4767*   4768*   4773*   4774*   4779*
                        4780*   4785*   4786*   4791*   4792*   4812*   4814*   5399*   5400*   5405*   5406*   5468*   5469*
                        5504*   5505*   5566*   5567*   5596*   5597*   5644*   5645*   5675*   5676*   5924*   5925*   5929*
                        5930*   6005*   6006*   6012*   6017*   6085*   6086*   6107*   6108*   6206*   6207*   6208    6217*
                        6218*   6219    6315*   6316*   6317    6326*   6327*   6328    6430*   6431*   6432    6441*   6442*
                        6443    6545*   6546*   6547    6556*   6557*   6558    6669*   6670*   6671    6680*   6681*   6682
                        6858*   6912*   6971*   6972*   6973    7898*   7899*   7960*   7961*   8171*   8458*   8459*   8554*
                        8555*   8641*   8642*   8745*   8746*   8807*   8808*   9719*   9720*   9723    9741*   9747*   9763*
                        9780*   9796*   9824*   9874*   9875*   9887*   9888*   9898*   9899*   9901    9919*   9920*   9921
                        9941*   9942*   9943    9974*   9975*   9976    9990*   9991*   9992   10004*  10005*  10017*  10018*
                       10034*  10035*  10060*  10061*  10075*  10076*  10090*  10091*  10105*  10106*  10118*  10119*  10148*
                       10149*  10163*  10164*  10178*  10179*  10195*  10196*  10221*  10222*  10236*  10237*  10251*  10252*
                       10266*  10267*  10289*  10290*  10304*  10305*  10326*  10327*  10345*  10366*  10417*  10418*  10419
                       10444*  10445*  10446   10460*  10461   10472*  10473   10590*  10617*  10618*  10619   10639*  10650*
                       10651*  10657   10668   10681*  10694*  10719*  10733*  10734*  10747*  10763*  10847*  10848*  10850
                       10860*  10869*  10877   10887*  10888*  10897*  10898*  10900   10910*  10919*  10920*  10922   10932*
                       10933*  10942*  10943*  10945   10987*  11001*  11017*  11031*  11045*  11059*  11073*  11128*  11142*
                       11158*  11175*  11196*  11213*  11227*  11252*  11267*  11282*  11298*  11315*  11333*  11347*  11363*
                       11396*  11397*  11399   11408*  11409*  11411   11421*  11430*  11437*  11440   11449*  11450*  11452
                       11461*  11462*  11464   11474*  11483*  11560*  11578*  11609*  11610*  11611   11631*  11632*  11635
                       11655*  11672*  11688*  11711*  11729*  11745*  11759*  11823*  11839*  11855*  11875*  11889*  11945*
                       11960*  12936
$BELL   001212          2045#   12458   12491   12542   12739
$BIN    062074         12068*  12071*  12072   12079#
$CDW1   001302          2096#
$CDW2   001304          2097#
$CHARC  063026         12269*  12279*  12286   12295*  12300#
$CKSWR  064232         12575#  12829
$CMTAG  001114          2008#   3922    3923    3931    3937    3938    3939
$CM3  = 000000          2038#
```

```
$CMM  = 000005           2038#   2039#   2040#   2041#   2042#   2043#
$CNTLC  065130          12530   12611   12739#
$CNTLG  065142          12596   12741#
$CNTLU  065135          12621   12714   12740#
$CPUOP  001250           2068#
$CRLF   001217           2047#   9059    9094    9127    9165    9175    9188    9196    9216    12268   12303   12466   12491
                        12632   12719   12739
$DBLK   062312          12104   12138   12146#
$DC40   001306           2098#
$DC41   001310           2099#
$DC42   001312           2100#
$DC43   001314           2101#
$DC44   001316           2102#
$DC45   001320           2103#
$DC46   001322           2104#
$DC47   001324           2105#
$DEVCT  001232           2059#
$DEVM   001300           2095#   3997*   4102*   4110*   4128*   4129    4136
$DOAGN  042234           9041    9062    9068#
$DTBL   062302          12107   12142#
$ENDAD  042224           1966    3968    9064#   12486
$ENDCT  042070           3937    9043#
$ENULL  042240           9071#
$ENV    001242           2064#   3974    3990    4163    12247   12468   12890   12914
$ENVM   001243           2065#   3960    3995    12249   12254   12892
$EOP    042034           4217    9033#   12003
$EOPCT  042062           3937*   9040#   9044
$EOSP   042000           4260    4296    4615    8928    9015#
$ERFLG  001117           2011#   12308   12348   12350   12356*  12378   12453*  12491
$ERMAX  001131           2017#   3940*   12350   12373*  12378
$ERROR  063534           3931    12451#
$ERRPC  001132           2018#   9119    12460*  12461*  12462   12491
$ERRTB  001562           2206#   9134
$ERTTL  001126           2015#   9056    9060*   12459*  12491
$ESCAP  001210           2044#   3939*   12372*  12482   12484   12491
$ETABL  001242           2063#
$ETEND  001326           1999    2106#
$FATAL  001224           2056#   12918*
$FFLG   066024          12881*  12884*  12912   12921*  12929#
$FILLC  001172           2036#   12272   12303
$FILLS  001171           2035#   12303
$GOAOR  001134           2019#   12936
$GOOAT  001140           2021#   4365*   4366    4370*   4379*   4380    4384*   4402*   4403    4407*   4414*   4415    4419*
                        4426*   4427    4545*   4583*   4584*   4585*   4593*   4601*   4607*   4611*   4612*   4620*   4704*
                        4712*   4718*   4724*   4730*   4736*   4755*   4763*   4769*   4775*   4781*   4787*   4793*   4813*
                        5401*   5407*   5467*   5506*   5568*   5598*   5646*   5674*   5923*   5931*   6008*   6013*   6018*
                        6087*   6109*   6204*   6205*   6208    6212*   6215*   6216*   6219    6313*   6314*   6317    6321*
                        6324*   6325*   6328    6428*   6429*   6432    6436*   6439*   6440*   6443    6543*   6544*   6547
                        6551*   6554*   6555*   6558    6667*   6668*   6671    6675*   6678*   6679*   6682    6857*   6911*
                        6970*   6973    7897*   7962*   8170*   8457*   8553*   8640*   8744*   8806*   9721*   9722*   9723
                        9739*   9740*   9746*   9748*   9760*   9761*   9762*   9778*   9779*   9794*   9795*   9822*   9823*
                        9876*   9889*   9900*   9901    9913*   9918*   9921    9932*   9940*   9943    9969*   9972*   9973*
                        9976    9988*   9989*   9992    9999*   10351*  10357*  10358*  10359*  10360*  10363*  10364   10411*
                       10416*  10419   10428*  10441*  10446   10457*  10458*  10459*  10461   10470*  10471*  10473   10589*
                       10601*  10614*  10619   10638*  10652*  10653*  10680*  10693*  10717*  10718*  10732*  10745*  10746*
                       10761*  10762*  10849*  10850   10859*  10875*  10877   10886*  10899*  10900   10909*  10921*  10922
```

```
                    10931* 10944* 10945  10985* 10986* 10999* 11000* 11018* 11019* 11032* 11033* 11046* 11047*
                    11060* 11061* 11074* 11075* 11126* 11127* 11140* 11141* 11156* 11157* 11173* 11174* 11194*
                    11195* 11211* 11212* 11225* 11226* 11250* 11251* 11266* 11268* 11280* 11281* 11296* 11297*
                    11313* 11314* 11331* 11332* 11345* 11346* 11361* 11362* 11398* 11399  11410* 11411  11420*
                    11451* 11452  11463* 11464  11472* 11558* 11559* 11576* 11577* 11592* 11605* 11611  11633*
                    11634* 11635  11639  11653* 11654* 11670* 11671* 11686* 11687* 11709* 11710* 11727* 11728*
                    11743* 11744* 11757* 11758* 11821* 11822* 11837* 11853* 11854* 11873* 11874* 11887* 11888*
                    11946* 11961* 12936
$GET42  042214       9061#
$GTSWR  064322      12597#  12827
$HD   = 000001       1505    1506
$HIBTS  001100       1994#
$HIOCT  065272      12779*  12784#
$ICNT   001120       2012#  12363*  12364   12366*  12377
$ILLUP  065542      12839   12855   12872#
$INTAG  001151       2026#  12594*  12613   12633   12746
$ITEMB  001130       2016#   9110   12462*  12470   12491
$LF     001220       2048#  12303   12491   12729   12739
$LFLG   066023      12922*  12928#
$LPADR  001122       2013#   3941*   4184*   4307*   4899*   4957*   5033*   5105*   5168*   7007*   7080*   7153*   7233*
                     7313*   7394*   7474*   8008*   8938*  12345*  12354*  12370*  12375   12377
$LPERR  001124       2014#   3942*   4185*   4308*   4900*   4958*   5034*   5106*   5169*   7008*   7081*   7154*   7234*
                     7314*   7395*   7475*   8009*   8939*  12354   12371*  12377   12481
$MADR1  001254       2081#
$MADR2  001260       2085#
$MADR3  001264       2088#
$MADR4  001270       2091#
$MAIL   001222       1995    1999    2054#   3959    3974    4163    4190    4230    4274    4313    4443    4520    4567
                     4633    4680    4827    4905    4963    5039    5111    5174    5241    5304    5367    5442    5541
                     5619    5702    5817    5891    5967    6053    6122    6237    6346    6461    6576    6700    6776
                     6870    6928    7013    7086    7159    7239    7319    7400    7480    7555    7651    7750    7849
                     7928    8014    8094    8195    8304    8417    8513    8607    8694    8774    8856    8944   12247
                    12369   12468
$MAMS1  001252       2075#
$MAMS2  001256       2083#
$MAMS3  001262       2086#
$MAMS4  001266       2089#
$MBADR  001102       1995#
$MFLG   066022      12882*  12888   12923*  12927#
$MNEW   065160      12600   12744#
$MSGAD  001236       2061#  12898*  12901
$MSGLG  001240       2062#  12903*
$MSGTY  001222       2055#  12896   12904*  12916   12920#
$MSWR   065147      12597   12742#
$MTYP1  001253       2076#
$MTYP2  001257       2084#
$MTYP3  001263       2087#
$MTYP4  001267       2090#
$MXCNT  063346      12367   12377#
$NULL   001170       2034#  12274   12303
$NWTST= 000001       4179#   4181    4220#   4222    4264#   4266    4299#   4301    4433#   4435    4510#   4512    4557#
                     4559    4624#   4671#   4817#   4819    4891#   4893    4950#   5026#   5098#   5161#   5231#   5233
                     5294#   5296    5357#   5359    5433#   5532#   5610#   5693#   5808#   5882#   5884    5957#   5959
                     6044#   6113#   6228#   6337#   6452#   6567#   6691#   6767#   6861#   6919#   6999#   7001    7072#
                     7074    7145#   7147    7225#   7227    7305#   7307    7386#   7388    7466#   7468    7546#   7642#
                     7741#   7840#   7919#   8001#   8085#   8186#   8295#   8408#   8504#   8591#   8685#   8765#   8847#
```

```
                      8931#
$OCNT   062544       12179* 12208* 12221#
$OMODE  062546       12174* 12178* 12183  12186* 12197* 12223#
$OVER   063332       12321  12346  12355  12365  12374#
$PASS   001230        2058#  3959*  9037*  9038*  9049   9071  12361  12378
$PASTM  001106        1997#
$POWER  065550       12870  12875#
$PWRON  065402        3935  12839#  12867
$PWRMG  065536       12870#
$PWRUP  065454       12849  12855#
$QUES   001216        2046# 12303  12491  12651  12722  12739
$RDCHR  064574       12664# 12830
$RDDEC= ****** U     12833
$RDLIN  064664       12687# 12831
$RDOCT  065172       12757# 12832
$RDSZ = 000010       12680#
$RESRE  061764       12044# 12834
$RTNAD  042236        9070#
$R2A  = ****** U     12835
$SAVRE  061726       12028# 12833
$SAVR6  065546       12848* 12856  12857* 12858* 12874#
$SCOPE  063032        3929  12317#
$SETUP= 000137        1948#  3928   3929   3931   3933   3935   3937   3938   3939   3941   3968   3971   4160
                      9035  12318  12452  12478  12486  12534  12539  12540  12570  12746
$STUP = 177777        1948#
$SVLAD  063276       12329  12368#
$SVPC = 000200        1964#  1969
$SWR  = 167400        1494#  1505   1510   1511   1512   1513   1514   1515   1516   1517   2043   2044   2045
                      3938   3939   3941   3942   4190   4230   4274   4313   4443   4520   4567   4633   4680
                      4827   4905   4963   5039   5111   5174   5241   5304   5367   5442   5541   5619   5702
                      5817   5891   5967   6053   6122   6237   6346   6461   6576   6700   6776   6870   6928
                      7013   7086   7159   7239   7319   7400   7480   7555   7651   7750   7849   7928   8014
                      8094   8195   8304   8417   8513   8600   8694   8774   8856   8944   9030   9036   9063
                      9069   9071  12309  12310  12311  12312  12313  12320  12332  12334  12335  12348  12349
                     12350  12357  12358  12359  12371  12374  12377  12443  12444  12445  12446  12447  12456
                     12463  12475  12479  12491  12871
$SWREG  001244        2066#  3962
$SWRMK= 000000        1517   1518  12313  12314  12338
$SWOBT  063350       12344  12378#
$TESTN  001226        2057#  4190*  4230*  4274*  4313*  4443*  4520*  4567*  4633*  4680*  4827*  4905*  4963*
                      5039*  5111*  5174*  5241*  5304*  5367*  5442*  5541*  5619*  5702*  5817*  5891*  5967*
                      6053*  6122*  6237*  6346*  6461*  6576*  6700*  6776*  6870*  6928*  7013*  7086*  7159*
                      7239*  7319*  7400*  7480*  7555*  7651*  7750*  7849*  7928*  8014*  8094*  8195*  8304*
                      8417*  8513*  8600*  8694*  8774*  8856*  8944*  9102  12369#
$TIMES  001206        2043#  3938*  4306*  4898*  4956*  5032*  5104*  5167*  7006*  7079*  7152*  7232*  7312*
                      7393*  7473*  8007*  8937*  9036* 12357* 12364  12367* 12377
$TKB    001162        2031# 12494  12515  12526  12551  12579  12606
$TKCNT  063734       12495# 12510* 12540  12557* 12671  12673#
$TKINT  063744        4003   4172   4354  12510# 12531  12592
$TKQEN= 063743       12499# 12565  12676
$TKQIN  063736       12496# 12511* 12512  12563# 12564# 12565  12567*
$TKQOU  063740       12497# 12512* 12674  12675* 12676  12678#
$TKQSR  063742       12498# 12511  12567  12678
$TKS    001160        2030# 12494  12516* 12547* 12549  12555* 12577  12593* 12603  12615* 12635*
$TKSRV  064014       12513  12526#
$TMPO   001174        2038#  4244*  4247   6913*  6975*  9497*  9570* 10525* 10526  12936
```

```
$TMP1   001176          2039#  4013#  4014   4015   4018   4027#  4028   4029   4037#  4038   4039   4049#  4051
                        4055   4056#  4057#  4059   4065#  4067   4071   4072#  4073#  4074#  4075#  4076#  4077#
                        4078#  4080   4086#  4088   4093#  4094#  4095#  4096#  4097#  4098   4106#  4107   4108
                        4114#  4115   4116   4118   4120   4124   4243#  4251   4293#  9498#  9499   9571#  9572
                       10524#  10531  11930#  11931#  11943  11945  12936
$TMP2   001200          2040#  4294#  11933#  11934#  11957  11960  12936
$TMP3   001202          2041#
$TMP4   001204          2042#
$TN     = 000073        1505#  4179   4190#  4220   4230#  4264   4274#  4299   4313#  4433   4443#  4510   4520#
                        4557   4567#  4624   4633#  4671   4680#  4817   4827#  4891   4905#  4950   4963#  5026
                        5039#  5098   5111#  5161   5174#  5231   5241#  5294   5304#  5357   5367#  5433   5442#
                        5532   5541#  5610   5619#  5693   5702#  5808   5817#  5882   5891#  5957   5967#  6044
                        6053#  6113   6122#  6228   6237#  6337   6346#  6452   6461#  6567   6576#  6691   6700#
                        6767   6776#  6861   6870#  6919   6928#  6999   7013#  7072   7086#  7145   7159#  7225
                        7239#  7305   7319#  7386   7400#  7466   7480#  7546   7555#  7642   7651#  7741   7750#
                        7840   7849#  7919   7928#  8001   8014#  8085   8094#  8186   8195#  8295   8304#  8408
                        8417#  8504   8513#  8591   8600#  8685   8694#  8765   8774#  8847   8856#  8931   8944#
                       12339  12379
$TPB    001166          2033#  12292#  12303
$TPFLG  001173          2037#  12241   12303
$TPS    001164          2032#  12290   12303
$TRAP   065274          3933   12793#
$TRAP2  065334         12808#  12819
$TRP    = 000016       12812#  12821#  12822#  12823#  12824#  12825#  12826#  12827  12828#  12829  12830#  12831#  12832#
                       12833#  12834#  12835#
$TRPAD  065346         12802   12819#
$TSTM   001104          1996#
$TSTNM  001116          2010#   9035#  12308  12341#  12368#  12369  12374  12378  12455  12491
$TTYIN  065120         12689  12690  12702  12720  12734  12738#
$TYPBN  062022         12065#  12825
$TYPOS  062076         12092#  12824
$TYPE   062550         12241#  12812  12820  12909
$TYPEC  062762         12271   12278  12285  12290#  12291  12637
$TYPEX  063030         12296   12298  12301#
$TYPOC  062346         12177#  12821
$TYPON  062362         12176   12179#  12823
$TYPOS  062322         12172#  12822
$UNIT   001234          2060#   4177#  9096   9721
$UNITM  001110          1998#
$USWR   001246          2067#
$VECT1  001272          2092#   4056   4071#  4072   4098#  5711   5717   5825   5831
$VECT2  001274          2093#
$XTSTR  063050         12323#
$$GET4= 000000          9063#
$$SW08= 000073         12378#  12379  12380#  12381#  12382#  12383#  12384#  12385#  12386#  12387#  12388#  12389#  12390#
                       12391#  12392#  12393#  12394#  12395#  12396#  12397#  12398#  12399#  12400#  12401#  12402#  12403#
                       12404#  12405#  12406#  12407#  12408#  12409#  12410#  12411#  12412#  12413#  12414#  12415#  12416#
                       12417#  12418#  12419#  12420#  12421#  12422#  12423#  12424#  12425#  12426#  12427#  12428#  12429#
                       12430#  12431#  12432#  12433#  12434#  12435#  12436#  12437#
$DFILL  062545         12173#  12177#  12187  12222#
$40CAT= ****** U       12320  12465
.       = 111321        1951#   1955#  1964   1965#  1967#  1969#  1974#  1977#  1983   1984#  1986#  1988#  2000
                        2007#   2049   2167#  2183#  2188#  3926   3941   3942   9048#  9071   9072#  12010#  12146#
                       12303  12377  12378  12491  12494  12498#  12499  12500#  12738#  12739  12746#  12851  12873
                       12930#  12936#
.$ASTA= ****** U       12882  12885
```

```
.$X    = 001100          1983#   1988
```