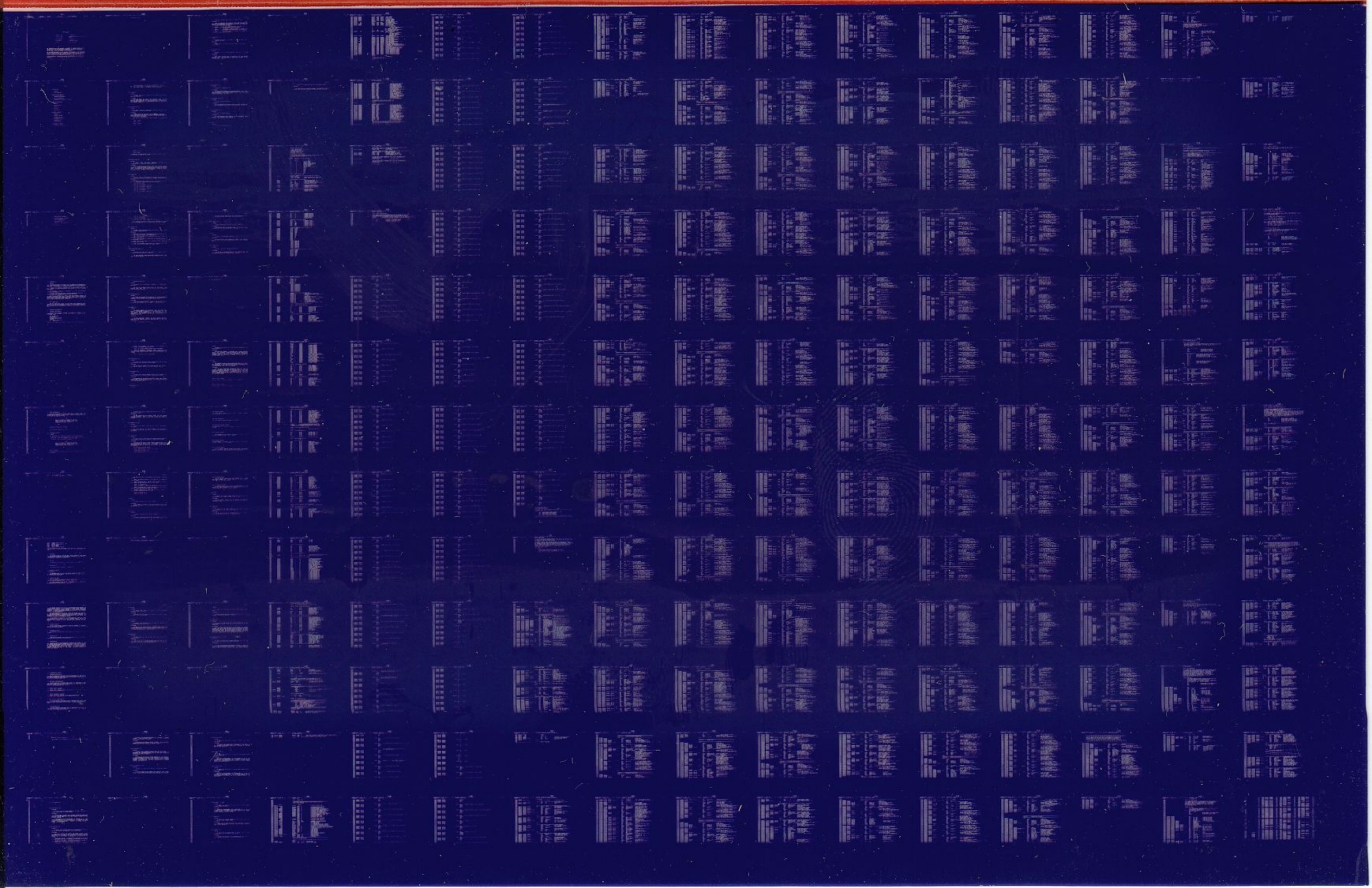
RM03,02

FUNCTIONAL TEST 1
CZRMCB0

AH-A998B-MC

FICHE 1 OF 2

JAN 1978
digital
MADE IN USA



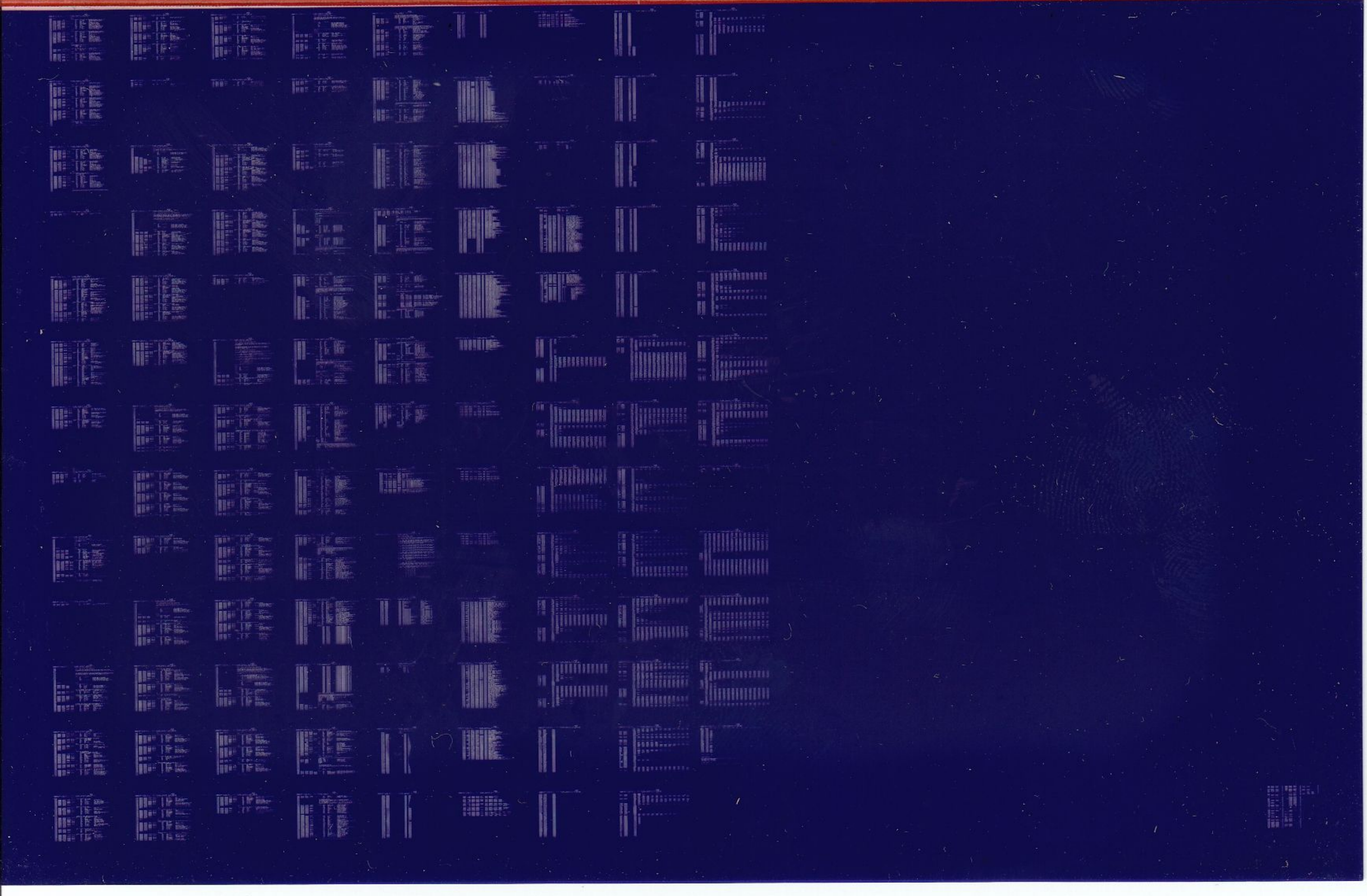
RM03,02

FUNCTIONAL TEST 1
CZRMCB0

AH-A998B-MC

FICHE 2 OF 2

JAN 1978
digital
MADE IN USA



.REM \

IDENTIFICATION

PRODUCT CODE:	AC-A9978-MC
PRODUCT NAME:	CZRMCB0 RM03/RM02 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 STORAGE MODULE DISK DRIVE. NOTE THAT A DISK PACK IS REQUIRED FOR TESTING AND IS CONSIDERED AN INTEGRAL OF THE STORAGE MODULE DISK DRIVE.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

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

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

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

182 DRIVES AND DISK PACKS.

2.2 MEDIA REQUIREMENTS

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

NOTE

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

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, CZRMJ-B

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

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

3.2 SWITCH OPTIONS

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

```
239                             SW15    HALT ON ERROR
240                             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.

404
405
406
407
408
409

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, F0-F4 AND GO=0, ALL OTHER BITS ARE DONT CARES;

.RMDS, IS CHECKED FOR 0, EXCEPT FOR 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

```
857                     RMDS IS ALSO SET.  ADDITIONALLY, CONTROLLER, ADAPTER,  AND  DRIVE
858                     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:

TO 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 (001777) 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:
```

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.

SEEK PRIME CYLINDERS

THIS TEST SEEKS FORWARD TO EACH PRIME CYLINDER ADDRESS, I.E.,CYLINDERS 1,2,4,8,...,512.

SEEK ZERO DIFFERENCE

THIS TEST EXECUTES SUCCESSIVE SEEKS TO CYLINDER 0.

SEEK MAXIMUM DIFFERENCE FORWARD

THIS TEST SEEKS TO CYLINDER 0 FOLLOWED BY A SEEK TO THE LAST CYLINDER.

SEEK ADJACENT FORWARD

THIS TEST SEEKS TO CYLINDER 0, FOLLOWED BY A SEEK TO THE ADJACENT FORWARD CYLINDER, I.E., CYLINDER 1.

SEEK ADJACENT REVERSE

THIS TEST SEEKS TO CYLINDER 1, FOLLOWED BY A SEEK TO THE ADJACENT REVERSE CYLINDER, I.E., CYLINDER 0.

SEEK TO INVALID SECTOR

WITH THE OFFSET REGISTER SET FOR 18 BIT FORMAT, THE PROGRAM SEEKS TO CYLINDER 0, TRACK 0, FOR EACH INVALID SECTOR ADDRESS AND 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  CZRMCB0 RM03/2 FCTNL TST 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
```

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

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

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

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

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

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

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

                                ;RMDA   DISK ADDRESS REGISTER

                                TA4     =       BIT10           ;TRACK ADDRESS 4
                                TA2     =       BIT09           ;TRACK ADDRESS 2
                                TA1     =       BIT08           ;TRACK ADDRESS 1
                                SA16    =       BIT04           ;SECTOR ADDRESS 16
                                SA8     =       BIT03           ;SECTOR ADDRESS 8
                                SA4     =       BIT02           ;SECTOR ADDRESS 4
                                SA2     =       BIT01           ;SECTOR ADDRESS 2
                                SA1     =       BIT00           ;SECTOR ADDRESS 1

                                ;TRACK,SECTOR MASKS

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

                                ;RMDS   DRIVE STATUS REGISTER

                                ATA     =       BIT15           ;ATTENTION ACTIVE
                                ERR     =       BIT14           ;COMPOSITE ERROR
                                PIP     =       BIT13           ;POSITIONING IN PROGRESS
                                MOL     =       BIT12           ;MEDIUM ON LINE
                                WRL     =       BIT11           ;WRITE LOCK
                                LBT     =       BIT10           ;LAST BLOCK TRANSFERRED
                                PGM     =       BIT09           ;PROGRAMMABLE
                                DPR     =       BIT08           ;DRIVE PRESENT
                                DRY     =       BIT07           ;DRIVE READY
                                VV      =       BIT06           ;VOLUME VALID
                                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
```

```
1829            000010                  BBO3    =       BIT03           ;TAG BUS
1830            000004                  BBO2    =       BIT02           ;TAG BUS
1831            000002                  BBO1    =       BIT01           ;TAG BUS
1832            000001                  BBO0    =       BIT00           ;TAG BUS
1833
1834
1835                                    ;RMER2  ERROR REGISTER 2
1836
1837            100000                  BSE     =       BIT15           ;BAD SECTOR ERROR
1838            040000                  SKI     =       BIT14           ;SEEK INCOMPLETE
1839            020000                  OPE     =       BIT13           ;OPERATOR PLUG ERROR
1840            010000                  IVC     =       BIT12           ;INVALID COMMAND ERROR
1841            004000                  LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
1842            002000                  LBC     =       BIT10           ;LOSS OF BIT CLOCK
1843            000200                  DVC     =       BIT07           ;DEVICE CHECK
1844            000010                  DPE     =       BIT03           ;DATA PARITY ERROR
1845
1846                                    .SBTTL  PROGRAM MNEMONICS
1847
1848            100000                  MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
1849            040000                  USE     =       BIT14           ;USER DETECTED SECTOR ERROR
1850
1851                                    .SBTTL  RM03 REGISTER INDEX VALUES
1852
1853            000000                  RMCS1   =       00              ;CONTROL STATUS REGISTER
1854            000006                  RMDA    =       06              ;DISK ADDRESS REGISTER
1855            000012                  RMDS    =       12              ;DRIVE STATUS REGISTER
1856            000014                  RMER1   =       14              ;ERROR REGISTER 1
1857            000016                  RMAS    =       16              ;ATTENTION SUMMARY REGISTER
1858            000020                  RMLA    =       20              ;LOOK AHEAD REGISTER
1859            000024                  RMMR1   =       24              ;MAINTENANCE REGISTER
1860            000026                  RMDT    =       26              ;DRIVE TYPE REGISTER
1861            000030                  RMSN    =       30              ;SERIAL NUMBER REGISTER
1862            000032                  RMOF    =       32              ;OFFSET REGISTER
1863            000034                  RMDC    =       34              ;DESIRED CYLINDER REGISTER
1864            000036                  RMCC    =       36              ;CURRENT CYLINDER REGISTER
1865            000040                  RMMR2   =       40              ;MAINTENANCE REGISTER 2
1866            000042                  RMER2   =       42              ;ERROR REGISTER 2
1867            000044                  RMEC1   =       44              ;ECC POSITION REGISTER
1868            000046                  RMEC2   =       46              ;ECC PATTERN REGISTER
1869            000050                  ILRG50  =       50              ;ILLEGAL REGISTER 50
1870            000052                  ILRG52  =       52              ;ILLEGAL REGISTER 52
1871            000054                  ILRG54  =       54              ;ILLEGAL REGISTER 54
1872            000056                  ILRG56  =       56              ;ILLEGAL REGISTER 56
1873            000060                  ILRG60  =       60              ;ILLEGAL REGISTER 60
1874            000062                  ILRG62  =       62              ;ILLEGAL REGISTER 62
1875            000064                  ILRG64  =       64              ;ILLEGAL REGISTER 64
1876            000066                  ILRG66  =       66              ;ILLEGAL REGISTER 66
1877            000070                  ILRG70  =       70              ;ILLEGAL REGISTER 70
1878            000072                  ILRG72  =       72              ;ILLEGAL REGISTER 72
1879            000074                  ILRG74  =       74              ;ILLEGAL REGISTER 74
1880            000076                  ILRG76  =       76              ;ILLEGAL REGISTER 76
1881
1882
1883            000077                  IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
1884
```

```
                        .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    042374                        $ENDAD                    ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
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                  001114  $CMTAG:                  ;;START OF COMMON TAGS
001114  000000                  .WORD   0
001116      000         001116  $TSTNM: .BYTE   0        ;;CONTAINS THE TEST NUMBER
001117      000         001117  $ERFLG: .BYTE   0        ;;CONTAINS ERROR FLAG
001120  000000          001120  $ICNT:  .WORD   0        ;;CONTAINS SUBTEST ITERATION COUNT
001122  000000          001122  $LPADR: .WORD   0        ;;CONTAINS SCOPE LOOP ADDRESS
001124  000000          001124  $LPERR: .WORD   0        ;;CONTAINS SCOPE RETURN FOR ERRORS
001126  000000          001126  $ERTTL: .WORD   0        ;;CONTAINS TOTAL ERRORS DETECTED
001130      000         001130  $ITEMB: .BYTE   0        ;;CONTAINS ITEM CONTROL BYTE
001131      001         001131  $ERMAX: .BYTE   1        ;;CONTAINS MAX. ERRORS PER TEST
001132  000000          001132  $ERRPC: .WORD   0        ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001134  000000          001134  $GDADR: .WORD   0        ;;CONTAINS ADDRESS OF 'GOOD' DATA
001136  000000          001136  $BDADR: .WORD   0        ;;CONTAINS ADDRESS OF 'BAD' DATA
001140  000000          001140  $GDDAT: .WORD   0        ;;CONTAINS 'GOOD' DATA
001142  000000          001142  $BDDAT: .WORD   0        ;;CONTAINS 'BAD' DATA
001144  000000                  .WORD   0        ;;RESERVED--NOT TO BE USED
001146  000000                  .WORD   0
001150      000         001150  $AUTOB: .BYTE   0        ;;AUTOMATIC MODE INDICATOR
001151      000         001151  $INTAG: .BYTE   0        ;;INTERRUPT MODE INDICATOR
001152  000000                  .WORD   0
001154  177570          001154  SWR:    .WORD   DSWR     ;;ADDRESS OF SWITCH REGISTER
001156  177570          001156  DISPLAY: .WORD  DDISP     ;;ADDRESS OF DISPLAY REGISTER
001160  177560          001160  $TKS:   177560           ;;TTY KBD STATUS
001162  177562          001162  $TKB:   177562           ;;TTY KBD BUFFER
001164  177564          001164  $TPS:   177564           ;;TTY PRINTER STATUS REG. ADDRESS
001166  177566          001166  $TPB:   177566           ;;TTY PRINTER BUFFER REG. ADDRESS
001170      000         001170  $NULL:  .BYTE   0        ;;CONTAINS NULL CHARACTER FOR FILLS
001171      002         001171  $FILLS: .BYTE   2        ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001172      012         001172  $FILLC: .BYTE   12       ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001173      000         001173  $TPFLG: .BYTE   0        ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001174  000000          001174  $TMP0:  .WORD   0        ;;USER DEFINED
001176  000000          001176  $TMP1:  .WORD   0        ;;USER DEFINED
001200  000000          001200  $TMP2:  .WORD   0        ;;USER DEFINED
001202  000000          001202  $TMP3:  .WORD   0        ;;USER DEFINED
001204  000000          001204  $TMP4:  .WORD   0        ;;USER DEFINED
001206  000000          001206  $TIMES: 0                ;;MAX. NUMBER OF ITERATIONS
001210  000000          001210  $ESCAPE:0                ;;ESCAPE ON ERROR ADDRESS
001212  177607  000377  001212  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001216      077         001216  $QUES:  .ASCII  /?/       ;;QUESTION MARK
001217      015         001217  $CRLF:  .ASCII  <15>      ;;CARRIAGE RETURN
001220  000012          001220  $LF:    .ASCIZ  <12>      ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001222                  001222  $MAIL:                   ;;APT MAILBOX
001222  000000          001222  $MSGTY: .WORD   AMSGTY    ;;MESSAGE TYPE CODE
001224  000000          001224  $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                                                        ;*
2071                                                        ;*
2072                                                        ;*
2073                                                        ;*
2074                                                        ;*
2075  001252      000               $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
2076  001253      000               $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
2077                                                        ;*
2078                                                        ;*
2079                                                        ;*
2080                                                        ;*
2081  001254  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
2082                                                        ;*
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:
```

MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE

MEM.TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003

BITS 15-11=CPU TYPE
11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
11/70=06,PDQ=07,Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT

```
                                ;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                   MEDENB: .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
```

```
2192                                    .SBTTL  ERROR POINTER TABLE
2193
2194                                    ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
2195                                    ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
2196                                    ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
2197                                    ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
2198                                    ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
2199
2200                                    ;*      EM              ;;POINTS TO THE ERROR MESSAGE
2201                                    ;*      DH              ;;POINTS TO THE DATA HEADER
2202                                    ;*      DT              ;;POINTS TO THE DATA
2203                                    ;*      DF              ;;POINTS TO THE DATA FORMAT
2204
2205
2206  001562                            $ERRTB:
2207
2208
```

```
2209
2210  001562  070012                    ;ERROR  1       WRONG UNIT SELECTED
2211  001564  074104                            EMT1
2212  001566  074230                            EHT1
2213  001570  074320                            EDT1
2214                                              EFT1
2215
2216                                    ;ERROR  2       DEVICE WENT UNAVAILABLE
2217  001572  070016                            EMT2
2218  001574  074104                            EHT1
2219  001576  074230                            EDT1
2220  001600  074320                            EFT1
2221
2222
2223                                    ;ERROR  3       DEVICE WENT NONEXISTENT
2224  001602  070024                            EMT3
2225  001604  074104                            EHT1
2226  001606  074230                            EDT1
2227  001610  074320                            EFT1
2228
2229
2230                                    ;ERROR  4       CONTROLLER NOT READY
2231  001612  070032                            EMT4
2232  001614  074104                            EHT1
2233  001616  074230                            EDT1
2234  001620  074320                            EFT1
2235
2236
2237                                    ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
2238  001622  070040                            EMT5
2239  001624  074104                            EHT1
2240  001626  074230                            EDT1
2241  001630  074320                            EFT1
2242
2243
2244                                    ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
2245  001632  070046                            EMT6
2246  001634  074104                            EHT1
2247  001636  074230                            EDT1
2248  001640  074320                            EFT1
2249
2250
2251                                    ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
2252  001642  070054                            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  070062                            EMT10
2260  001654  074104                            EHT1
2261  001656  074230                            EDT1
2262  001660  074320                            EFT1
2263
2264
```

```
2265
2266   001662  070066                ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
2267   001664  074104                        EMT11
2268   001666  074230                        EHT1
2269   001670  074320                        EDT1
2270                                         EFT1
2271
2272                                 ;ERROR  12      INCORRECT FUNCTION CODE
2273   001672  070072                        EMT12
2274   001674  074104                        EHT1
2275   001676  074230                        EDT1
2276   001700  074320                        EFT1
2277
2278
2279                                 ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
2280   001702  070100                        EMT13
2281   001704  074104                        EHT1
2282   001706  074230                        EDT1
2283   001710  074320                        EFT1
2284
2285
2286                                 ;ERROR  14      TRANSFER ERROR IS INCORRECT
2287   001712  070112                        EMT14
2288   001714  074104                        EHT1
2289   001716  074230                        EDT1
2290   001720  074320                        EFT1
2291
2292
2293                                 ;ERROR  15      INCORRECT WORD COUNT
2294   001722  070120                        EMT15
2295   001724  074104                        EHT1
2296   001726  074230                        EDT1
2297   001730  074320                        EFT1
2298
2299
2300                                 ;ERROR  16      INCORRECT BUS ADDRESS
2301   001732  070126                        EMT16
2302   001734  074104                        EHT1
2303   001736  074230                        EDT1
2304   001740  074320                        EFT1
2305
2306
2307                                 ;ERROR  17      INCORRECT LBT STATUS
2308   001742  070136                        EMT17
2309   001744  074104                        EHT1
2310   001746  074230                        EDT1
2311   001750  074320                        EFT1
2312
2313
2314                                 ;ERROR  20      INCORRECT AOE
2315   001752  070146                        EMT20
2316   001754  074104                        EHT1
2317   001756  074230                        EDT1
2318   001760  074320                        EFT1
2319
2320
```

```
2321
2322  001762  070156                ;ERROR  21      INCORRECT DISK ADDRESS
2323  001764  074104                        EMT21
2324  001766  074230                        EHT1
2325  001770  074320                        EDT1
2326                                  EFT1
2327
2328                                ;ERROR  22      INCORRECT CYLINDER ADDRESS
2329  001772  070166                        EMT22
2330  001774  074104                        EHT1
2331  001776  074230                        EDT1
2332  002000  074320                        EFT1
2333
2334
2335                                ;ERROR  23      INCORRECT WLE STATUS
2336  002002  070176                        EMT23
2337  002004  074104                        EHT1
2338  002006  074230                        EDT1
2339  002010  074320                        EFT1
2340
2341
2342                                ;ERROR  24      INCORRECT UPE STATUS
2343  002012  070206                        EMT24
2344  002014  074104                        EHT1
2345  002016  074230                        EDT1
2346  002020  074320                        EFT1
2347
2348
2349                                ;ERROR  25      INCORRECT WCF STATUS
2350  002022  070216                        EMT25
2351  002024  074104                        EHT1
2352  002026  074230                        EDT1
2353  002030  074320                        EFT1
2354
2355
2356                                ;ERROR  26      INCORRECT WCE STATUS
2357  002032  070226                        EMT26
2358  002034  074104                        EHT1
2359  002036  074230                        EDT1
2360  002040  074320                        EFT1
2361
2362
2363                                ;ERROR  27      INCORRECT MDPE STATUS
2364  002042  070236                        EMT27
2365  002044  074104                        EHT1
2366  002046  074230                        EDT1
2367  002050  074320                        EFT1
2368
2369
2370                                ;ERROR  30      INCORRECT DCK STATUS
2371  002052  070246                        EMT30
2372  002054  074104                        EHT1
2373  002056  074230                        EDT1
2374  002060  074320                        EFT1
2375
2376
```

```
2377
2378  002062  070256          ;ERROR  31      INCORRECT ECH STATUS
2379  002064  074104                  EMT31
2380  002066  074230                  EHT1
2381  002070  074320                  EDT1
2382                                   EFT1
2383
2384                          ;ERROR  32      DLT SHOULD NOT BE SET
2385  002072  070266                  EMT32
2386  002074  074104                  EHT1
2387  002076  074230                  EDT1
2388  002100  074320                  EFT1
2389
2390
2391                          ;ERROR  33      MXF SHOULD NOT BE SET
2392  002102  070276                  EMT33
2393  002104  074104                  EHT1
2394  002106  074230                  EDT1
2395  002110  074320                  EFT1
2396
2397
2398                          ;ERROR  34      DTE SHOULD NOT BE SET
2399  002112  070306                  EMT34
2400  002114  074104                  EHT1
2401  002116  074230                  EDT1
2402  002120  074320                  EFT1
2403
2404
2405                          ;ERROR  35      INCORRECT HCRC STATUS
2406  002122  070316                  EMT35
2407  002124  074104                  EHT1
2408  002126  074230                  EDT1
2409  002130  074320                  EFT1
2410
2411
2412                          ;ERROR  36      INCORRECT HCE STATUS
2413  002132  070326                  EMT36
2414  002134  074104                  EHT1
2415  002136  074230                  EDT1
2416  002140  074320                  EFT1
2417
2418
2419                          ;ERROR  37      INCORRECT FER STATUS
2420  002142  070336                  EMT37
2421  002144  074104                  EHT1
2422  002146  074230                  EDT1
2423  002150  074320                  EFT1
2424
2425
2426                          ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
2427  002152  070346                  EMT40
2428  002154  074104                  EHT1
2429  002156  074230                  EDT1
2430  002160  074320                  EFT1
2431
2432
```

```
2433
2434   002162  070354          ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
2435   002164  074104                  EMT41
2436   002166  074230                  EHT1
2437   002170  074320                  EDT1
2438                                    EFT1
2439
2440                           ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
2441   002172  070364                  EMT42
2442   002174  074104                  EHT1
2443   002176  074230                  EDT1
2444   002200  074320                  EFT1
2445
2446
2447                           ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
2448   002202  070376                  EMT43
2449   002204  074104                  EHT1
2450   002206  074230                  EDT1
2451   002210  074320                  EFT1
2452
2453
2454                           ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
2455   002212  070406                  EMT44
2456   002214  074104                  EHT1
2457   002216  074230                  EDT1
2458   002220  074320                  EFT1
2459
2460
2461                           ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
2462   002222  070416                  EMT45
2463   002224  074104                  EHT1
2464   002226  074230                  EDT1
2465   002230  074320                  EFT1
2466
2467
2468                           ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
2469   002232  070426                  EMT46
2470   002234  074104                  EHT1
2471   002236  074230                  EDT1
2472   002240  074320                  EFT1
2473
2474
2475                           ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
2476   002242  070436                  EMT47
2477   002244  074104                  EHT1
2478   002246  074230                  EDT1
2479   002250  074320                  EFT1
2480
2481
2482                           ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
2483   002252  070444                  EMT50
2484   002254  074104                  EHT1
2485   002256  074230                  EDT1
2486   002260  074320                  EFT1
2487
2488
```

```
2489                                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
2490   002262  070454                           EMT51
2491   002264  074104                           EHT1
2492   002266  074230                           EDT1
2493   002270  074320                           EFT1
2494
2495
2496                                    ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
2497   002272  070466                           EMT52
2498   002274  074104                           EHT1
2499   002276  074230                           EDT1
2500   002300  074320                           EFT1
2501
2502
2503                                    ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
2504                                    ;               ON CYLINDER LATCH DIDN'T RESET
2505   002302  070504                           EMT53
2506   002304  074104                           EHT1
2507   002306  074230                           EDT1
2508   002310  074320                           EFT1
2509
2510
2511                                    ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
2512   002312  070522                           EMT54
2513   002314  074104                           EHT1
2514   002316  074230                           EDT1
2515   002320  074320                           EFT1
2516
2517
2518                                    ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
2519   002322  070532                           EMT55
2520   002324  074104                           EHT1
2521   002326  074230                           EDT1
2522   002330  074320                           EFT1
2523
2524
2525                                    ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
2526   002332  070544                           EMT56
2527   002334  074104                           EHT1
2528   002336  074230                           EDT1
2529   002340  074320                           EFT1
2530
2531
2532                                    ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
2533   002342  070562                           EMT57
2534   002344  074104                           EHT1
2535   002346  074230                           EDT1
2536   002350  074320                           EFT1
2537
2538
2539                                    ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
2540                                    ;               VOLUME VALID
2541   002352  070572                           EMT60
2542   002354  074104                           EHT1
2543   002356  074230                           EDT1
2544   002360  074320                           EFT1
```

```
                                        ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
                                        ;               VOLUME VALID IS STIL SET
002362  070610                                  EMT61
002364  074104                                  EHT1
002366  074230                                  EDT1
002370  074320                                  EFT1

                                        ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
                                        ;               REPORTED DURING SEEK COMMAND
002372  070630                                  EMT62
002374  074104                                  EHT1
002376  074230                                  EDT1
002400  074320                                  EFT1

                                        ;ERROR  63      UNUSED
002402  000000                                  0
002404  000000                                  0
002406  000000                                  0
002410  000000                                  0

                                        ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
002412  070646                                  EMT64
002414  074104                                  EHT1
002416  074230                                  EDT1
002420  074320                                  EFT1

                                        ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
002422  070666                                  EMT65
002424  074104                                  EHT1
002426  074230                                  EDT1
002430  074320                                  EFT1

                                        ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
002432  070706                                  EMT66
002434  074104                                  EHT1
002436  074230                                  EDT1
002440  074320                                  EFT1

                                        ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
002442  070720                                  EMT67
002444  074104                                  EHT1
002446  074230                                  EDT1
002450  074320                                  EFT1

                                        ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
002452  070732                                  EMT70
002454  074104                                  EHT1
```

```
2601   002456  074230                  EDT1
2602   002460  074320                  EFT1
2603
2604
2605                           ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
2606   002462  070742                  EMT71
2607   002464  074104                  EHT1
2608   002466  074230                  EDT1
2609   002470  074320                  EFT1
2610
2611
2612                           ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
2613   002472  070752                  EMT72
2614   002474  074104                  EHT1
2615   002476  074230                  EDT1
2616   002500  074320                  EFT1
2617
2618
2619                           ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
2620                           ;               CYLINDER DIDNT DROP
2621   002502  070770                  EMT73
2622   002504  074104                  EHT1
2623   002506  074230                  EDT1
2624   002510  074320                  EFT1
2625
2626
2627                           ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
2628   002512  071006                  EMT74
2629   002514  074104                  EHT1
2630   002516  074230                  EDT1
2631   002520  074320                  EFT1
2632
2633
2634                           ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
2635   002522  071016                  EMT75
2636   002524  074104                  EHT1
2637   002526  074230                  EDT1
2638   002530  074320                  EFT1
2639
2640
2641                           ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
2642   002532  071036                  EMT76
2643   002534  074104                  EHT1
2644   002536  074230                  EDT1
2645   002540  074320                  EFT1
2646
2647
2648                           ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
2649   002542  071046                  EMT77
2650   002544  074104                  EHT1
2651   002546  074230                  EDT1
2652   002550  074320                  EFT1
2653
2654
2655                           ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
2656   002552  071060                  EMT100
```

```
 2657  002554  074104                    EHT1
 2658  002556  074230                    EDT1
 2659  002560  074320                    EFT1
 2660
 2661
 2662                          ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
 2663  002562  071076                    EMT101
 2664  002564  074104                    EHT1
 2665  002566  074230                    EDT1
 2666  002570  074320                    EFT1
 2667
 2668
 2669                          ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
 2670  002572  071114                    EMT102
 2671  002574  074104                    EHT1
 2672  002576  074230                    EDT1
 2673  002600  074320                    EFT1
 2674
 2675
 2676                          ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
 2677  002602  071124                    EMT103
 2678  002604  074104                    EHT1
 2679  002606  074230                    EDT1
 2680  002610  074320                    EFT1
 2681
 2682
 2683                          ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
 2684  002612  071136                    EMT104
 2685  002614  074104                    EHT1
 2686  002616  074230                    EDT1
 2687  002620  074320                    EFT1
 2688
 2689
 2690                          ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
 2691  002622  071154                    EMT105
 2692  002624  074104                    EHT1
 2693  002626  074230                    EDT1
 2694  002630  074320                    EFT1
 2695
 2696
 2697                          ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
 2698  002632  071164                    EMT106
 2699  002634  074104                    EHT1
 2700  002636  074230                    EDT1
 2701  002640  074320                    EFT1
 2702
 2703
 2704                          ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
 2705  002642  071174                    EMT107
 2706  002644  074104                    EHT1
 2707  002646  074230                    EDT1
 2708  002650  074320                    EFT1
 2709
 2710
 2711                          ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
 2712  002652  071214                    EMT110
```

```
                                        EMT110
                                        EDT110
                                        EFT110

                        ;ERROR  111     NONEXISTENT DEVICE
002662  071226                          EMT111
002664  074132                          EHT111
002666  074250                          EDT111
002670  074340                          EFT111

                        ;ERROR  112     DEVICE NOT AVAILABLE
002672  071234                          EMT112
002674  074132                          EHT111
002676  074250                          EDT111
002700  074340                          EFT111

                        ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
002702  071242                          EMT113
002704  000000                          0
002706  000000                          0
002710  000000                          0

                        ;ERROR  114     DEVICE NOT AN RM03
002712  071256                          EMT114
002714  074136                          EHT114
002716  074252                          EDT114
002720  074342                          EFT114

                        ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
002722  071264                          EMT115
002724  074104                          EHT1
002726  074230                          EDT1
002730  074320                          EFT1

                        ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
002732  071274                          EMT116
002734  074104                          EHT1
002736  074230                          EDT1
002740  074320                          EFT1

                        ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
002742  071304                          EMT117
002744  074104                          EHT1
002746  074230                          EDT1
002750  074320                          EFT1

                        ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
002752  071314                          EMT120
```

```
2769  002754  074104                        EHT1
2770  002756  074230                        EDT1
2771  002760  074320                        EFT1
2772
2773
2774                          ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
2775  002762  071324                        EMT121
2776  002764  074104                        EHT1
2777  002766  074230                        EDT1
2778  002770  074320                        EFT1
2779
2780
2781                          ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
2782  002772  071334                        EMT122
2783  002774  074104                        EHT1
2784  002776  074230                        EDT1
2785  003000  074320                        EFT1
2786
2787
2788                          ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
2789  003002  071344                        EMT123
2790  003004  074104                        EHT1
2791  003006  074230                        EDT1
2792  003010  074320                        EFT1
2793
2794
2795                          ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
2796  003012  071354                        EMT124
2797  003014  074104                        EHT1
2798  003016  074230                        EDT1
2799  003020  074320                        EFT1
2800
2801
2802                          ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
2803  003022  071364                        EMT125
2804  003024  074104                        EHT1
2805  003026  074230                        EDT1
2806  003030  074320                        EFT1
2807
2808
2809                          ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2810  003032  071374                        EMT126
2811  003034  074104                        EHT1
2812  003036  074230                        EDT1
2813  003040  074320                        EFT1
2814
2815
2816                          ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
2817  003042  071406                        EMT127
2818  003044  074104                        EHT1
2819  003046  074230                        EDT1
2820  003050  074320                        EFT1
2821
2822
2823                          ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2824  003052  071420                        EMT130
```

```
2825  003054  074104                    EHT1
2826  003056  074230                    EDT1
2827  003060  074320                    EFT1
2828
2829
2830                          ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2831  003062  071432                    EMT131
2832  003064  074104                    EHT1
2833  003066  074230                    EDT1
2834  003070  074320                    EFT1
2835
2836
2837                          ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2838  003072  071444                    EMT132
2839  003074  074104                    EHT1
2840  003076  074230                    EDT1
2841  003100  074320                    EFT1
2842
2843
2844                          ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2845  003102  071456                    EMT133
2846  003104  074104                    EHT1
2847  003106  074230                    EDT1
2848  003110  074320                    EFT1
2849
2850
2851                          ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2852  003112  071470                    EMT134
2853  003114  074104                    EHT1
2854  003116  074230                    EDT1
2855  003120  074320                    EFT1
2856
2857
2858                          ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2859  003122  071502                    EMT135
2860  003124  074104                    EHT1
2861  003126  074230                    EDT1
2862  003130  074320                    EFT1
2863
2864
2865                          ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2866  003132  071514                    EMT136
2867  003134  074104                    EHT1
2868  003136  074230                    EDT1
2869  003140  074320                    EFT1
2870
2871
2872                          ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2873  003142  071526                    EMT137
2874  003144  074104                    EHT1
2875  003146  074230                    EDT1
2876  003150  074320                    EFT1
2877
2878
2879                          ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2880  003152  071536                    EMT140
```

```
2881   003154  074104                          EHT1
2882   003156  074230                          EDT1
2883   003160  074320                          EFT1
2884
2885
2886                                   ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
2887   003162  071546                          EMT141
2888   003164  074104                          EHT1
2889   003166  074230                          EDT1
2890   003170  074320                          EFT1
2891
2892
2893                                   ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
2894   003172  071556                          EMT142
2895   003174  074104                          EHT1
2896   003176  074230                          EDT1
2897   003200  074320                          EFT1
2898
2899
2900                                   ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
2901   003202  071566                          EMT143
2902   003204  074104                          EHT1
2903   003206  074230                          EDT1
2904   003210  074320                          EFT1
2905
2906
2907                                   ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
2908   003212  071576                          EMT144
2909   003214  074104                          EHT1
2910   003216  074230                          EDT1
2911   003220  074320                          EFT1
2912
2913
2914                                   ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
2915   003222  071606                          EMT145
2916   003224  074104                          EHT1
2917   003226  074230                          EDT1
2918   003230  074320                          EFT1
2919
2920
2921                                   ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
2922   003232  071616                          EMT146
2923   003234  074104                          EHT1
2924   003236  074230                          EDT1
2925   003240  074320                          EFT1
2926
2927
2928                                   ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
2929   003242  071626                          EMT147
2930   003244  074104                          EHT1
2931   003246  074230                          EDT1
2932   003250  074320                          EFT1
2933
2934
2935                                   ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
2936   003252  071636                          EMT150
```

```
2937  003254  074104                    EHT1
2938  003256  074230                    EDT1
2939  003260  074320                    EFT1
2940
2941
2942                              ;ERROR  151     MEDIUM NOT ON LINE
2943  003262  071646                    EMT151
2944  003264  074104                    EHT1
2945  003266  074230                    EDT1
2946  003270  074320                    EFT1
2947
2948
2949                              ;ERROR  152     DRIVE FAULT
2950  003272  071660                    EMT152
2951  003274  074104                    EHT1
2952  003276  074230                    EDT1
2953  003300  074320                    EFT1
2954
2955
2956                              ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2957  003302  071672                    EMT153
2958  003304  074104                    EHT1
2959  003306  074230                    EDT1
2960  003310  074320                    EFT1
2961
2962
2963                              ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2964  003312  071710                    EMT154
2965  003314  074104                    EHT1
2966  003316  074230                    EDT1
2967  003320  074320                    EFT1
2968
2969
2970                              ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2971  003322  071726                    EMT155
2972  003324  074104                    EHT1
2973  003326  074230                    EDT1
2974  003330  074320                    EFT1
2975
2976
2977                              ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2978  003332  071740                    EMT156
2979  003334  074104                    EHT1
2980  003336  074230                    EDT1
2981  003340  074320                    EFT1
2982
2983
2984                              ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2985  003342  071752                    EMT157
2986  003344  074104                    EHT1
2987  003346  074230                    EDT1
2988  003350  074320                    EFT1
2989
2990
2991                              ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2992  003352  071764                    EMT160
```

```
2993   003354  074104                  EHT1
2994   003356  074230                  EDT1
2995   003360  074320                  EFT1
2996
2997
2998                           ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2999   003362  071774                  EMT161
3000   003364  074104                  EHT1
3001   003366  074230                  EDT1
3002   003370  074320                  EFT1
3003
3004
3005                           ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
3006   003372  072006                  EMT162
3007   003374  074104                  EHT1
3008   003376  074230                  EDT1
3009   003400  074320                  EFT1
3010
3011
3012                           ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
3013                           ;               WRITE BUFFER
3014   003402  073700                  EMT336
3015   003404  074166                  EHT336
3016   003406  074264                  EDT336
3017   003410  074354                  EFT336
3018
3019
3020                           ;ERROR  164     OPI SHOULD NOT BE SET
3021   003412  072030                  EMT164
3022   003414  074104                  EHT1
3023   003416  074230                  EDT1
3024   003420  074320                  EFT1
3025
3026
3027                           ;ERROR  165     IVC SHOULD NOT BE SET
3028   003422  072036                  EMT165
3029   003424  074104                  EHT1
3030   003426  074230                  EDT1
3031   003430  074320                  EFT1
3032
3033
3034                           ;ERROR  166     IAE SHOULD NOT BE SET
3035   003432  072044                  EMT166
3036   003434  074104                  EHT1
3037   003436  074230                  EDT1
3038   003440  074320                  EFT1
3039
3040
3041                           ;ERROR  167     NEM SHOULD NOT BE SET
3042   003442  072052                  EMT167
3043   003444  074104                  EHT1
3044   003446  074230                  EDT1
3045   003450  074320                  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  072070                    EMT171
3057  003464  074104                    EHT1
3058  003466  074230                    EDT1
3059  003470  074320                    EFT1
3060
3061
3062                          ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
3063  003472  072100                    EMT172
3064  003474  074104                    EHT1
3065  003476  074230                    EDT1
3066  003500  074320                    EFT1
3067
3068
3069                          ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
3070  003502  072110                    EMT173
3071  003504  074104                    EHT1
3072  003506  074230                    EDT1
3073  003510  074320                    EFT1
3074
3075
3076                          ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
3077  003512  072120                    EMT174
3078  003514  074104                    EHT1
3079  003516  074230                    EDT1
3080  003520  074320                    EFT1
3081
3082
3083                          ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
3084  003522  072132                    EMT175
3085  003524  074104                    EHT1
3086  003526  074230                    EDT1
3087  003530  074320                    EFT1
3088
3089
3090                          ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
3091  003532  072140                    EMT176
3092  003534  074104                    EHT1
3093  003536  074230                    EDT1
3094  003540  074320                    EFT1
3095
3096
3097                          ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
3098  003542  072146                    EMT177
3099  003544  074104                    EHT1
3100  003546  074230                    EDT1
3101  003550  074320                    EFT1
3102
3103
3104                          ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
3105  003552  072160                    EMT200
3106  003554  074104                    EHT1
3107  003556  074230                    EDT1
3108  003560  074320                    EFT1
3109
3110
3111                          ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
3112  003562  072172                    EMT201
3113  003564  074104                    EHT1
3114  003566  074230                    EDT1
3115  003570  074320                    EFT1
3116
3117
3118                          ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
3119  003572  072204                    EMT202
3120  003574  074104                    EHT1
3121  003576  074230                    EDT1
3122  003600  074320                    EFT1
3123
3124
3125                          ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
3126  003602  072216                    EMT203
3127  003604  074104                    EHT1
3128  003606  074230                    EDT1
3129  003610  074320                    EFT1
3130
3131
3132                          ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
3133  003612  072230                    EMT204
3134  003614  074104                    EHT1
3135  003616  074230                    EDT1
3136  003620  074320                    EFT1
3137
3138
3139                          ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
3140  003622  072246                    EMT205
3141  003624  074104                    EHT1
3142  003626  074230                    EDT1
3143  003630  074320                    EFT1
3144
3145
3146                          ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
3147  003632  072256                    EMT206
3148  003634  074104                    EHT1
3149  003636  074230                    EDT1
3150  003640  074320                    EFT1
3151
3152
3153                          ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
3154  003642  072266                    EMT207
3155  003644  074104                    EHT1
3156  003646  074230                    EDT1
3157  003650  074320                    EFT1
3158
3159
3160                          ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
3161  003652  072300                        EMT210
3162  003654  074104                        EHT1
3163  003656  074230                        EDT1
3164  003660  074320                        EFT1
3165
3166
3167                          ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
3168  003662  072306                        EMT211
3169  003664  074104                        EHT1
3170  003666  074230                        EDT1
3171  003670  074320                        EFT1
3172
3173
3174                          ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
3175  003672  072322                        EMT212
3176  003674  074104                        EHT1
3177  003676  074230                        EDT1
3178  003700  074320                        EFT1
3179
3180
3181                          ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
3182  003702  072336                        EMT213
3183  003704  074104                        EHT1
3184  003706  074230                        EDT1
3185  003710  074320                        EFT1
3186
3187
3188                          ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
3189  003712  072346                        EMT214
3190  003714  074116                        EHT2
3191  003716  074240                        EDT2
3192  003720  074330                        EFT2
3193
3194
3195                          ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
3196  003722  072366                        EMT215
3197  003724  074116                        EHT2
3198  003726  074240                        EDT2
3199  003730  074330                        EFT2
3200
3201
3202                          ;ERROR  216     INCORRECT "IVC" STATUS
3203  003732  072400                        EMT216
3204  003734  074104                        EHT1
3205  003736  074230                        EDT1
3206  003740  074320                        EFT1
3207
3208
3209                          ;ERROR  217     INCORRECT "IAE" STATUS
3210  003742  072410                        EMT217
3211  003744  074104                        EHT1
3212  003746  074230                        EDT1
3213  003750  074320                        EFT1
3214
3215
3216                          ;ERROR 220      INCORRECT "WLE" STATUS
```

```
3217  003752  072420                    EMT220
3218  003754  074104                    EHT1
3219  003756  074230                    EDT1
3220  003760  074320                    EFT1
3221
3222
3223                              ;ERROR  221     INCORRECT "OPI" STATUS
3224  003762  072430                    EMT221
3225  003764  074104                    EHT1
3226  003766  074230                    EDT1
3227  003770  074320                    EFT1
3228
3229
3230                              ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
3231  003772  072440                    EMT222
3232  003774  074104                    EHT1
3233  003776  074230                    EDT1
3234  004000  074320                    EFT1
3235
3236
3237                              ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
3238  004002  072450                    EMT223
3239  004004  074142                    EHT223
3240  004006  074254                    EDT223
3241  004010  074344                    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  072524                    EMT246
3372  004234  074104                    EHT1
3373  004236  074230                    EDT1
3374  004240  074320                    EFT1
3375
3376
3377                            ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
3378  004242  072534                    EMT247
3379  004244  074104                    EHT1
3380  004246  074230                    EDT1
3381  004250  074320                    EFT1
3382
3383
3384                            ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
3385  004252  072546                        EMT250
3386  004254  074104                        EHT1
3387  004256  074230                        EDT1
3388  004260  074320                        EFT1
3389
3390
3391                          ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
3392  004262  072562                        EMT251
3393  004264  074116                        EHT2
3394  004266  074240                        EDT2
3395  004270  074330                        EFT2
3396
3397
3398                          ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
3399  004272  072570                        EMT252
3400  004274  074116                        EHT2
3401  004276  074240                        EDT2
3402  004300  074330                        EFT2
3403
3404
3405                          ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
3406  004302  072576                        EMT253
3407  004304  074104                        EHT1
3408  004306  074230                        EDT1
3409  004310  074320                        EFT1
3410
3411
3412                          ;ERROR  254     INCORRECT "ILF" STATUS
3413  004312  072614                        EMT254
3414  004314  074104                        EHT1
3415  004316  074230                        EDT1
3416  004320  074320                        EFT1
3417
3418
3419                          ;ERROR  255     INCORRECT "ATA" STATUS
3420  004322  072624                        EMT255
3421  004324  074104                        EHT1
3422  004326  074230                        EDT1
3423  004330  074320                        EFT1
3424
3425
3426                          ;ERROR  256     INCORRECT "ILR" STATUS
3427  004332  072634                        EMT256
3428  004334  074154                        EHT256
3429  004336  074254                        EDT223
3430  004340  074344                        EFT223
3431
3432
3433                          ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
3434  004342  072644                        EMT257
3435  004344  074104                        EHT1
3436  004346  074230                        EDT1
3437  004350  074320                        EFT1
3438
3439
3440                          ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
3441  004352  072656                    EMT260
3442  004354  074104                    EHT1
3443  004356  074230                    EDT1
3444  004360  074320                    EFT1
3445
3446
3447                              ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
3448  004362  072670                    EMT261
3449  004364  074104                    EHT1
3450  004366  074230                    EDT1
3451  004370  074320                    EFT1
3452
3453
3454                              ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
3455  004372  072710                    EMT262
3456  004374  074104                    EHT1
3457  004376  074230                    EDT1
3458  004400  074320                    EFT1
3459
3460
3461                              ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
3462  004402  072720                    EMT263
3463  004404  074104                    EHT1
3464  004406  074230                    EDT1
3465  004410  074320                    EFT1
3466
3467
3468                              ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
3469  004412  072730                    EMT264
3470  004414  074104                    EHT1
3471  004416  074230                    EDT1
3472  004420  074320                    EFT1
3473
3474
3475                              ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
3476  004422  072750                    EMT265
3477  004424  074104                    EHT1
3478  004426  074230                    EDT1
3479  004430  074320                    EFT1
3480
3481
3482                              ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
3483  004432  072770                    EMT266
3484  004434  074104                    EHT1
3485  004436  074230                    EDT1
3486  004440  074320                    EFT1
3487
3488
3489                              ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
3490                              ;               ADDRESS IS TOO LARGE
3491  004442  073002                    EMT267
3492  004444  074104                    EHT1
3493  004446  074230                    EDT1
3494  004450  074320                    EFT1
3495
3496
```

```
3497
3498  004452  073020                ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
3499  004454  074104                        EMT270
3500  004456  074230                        EHT1
3501  004460  074320                        EDT1
3502                                        EFT1
3503
3504                                ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
3505                                ;               DIDN'T DROP
3506  004462  073034                        EMT271
3507  004464  074104                        EHT1
3508  004466  074230                        EDT1
3509  004470  074320                        EFT1
3510
3511
3512                                ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
3513  004472  073052                        EMT272
3514  004474  074104                        EHT1
3515  004476  074230                        EDT1
3516  004500  074320                        EFT1
3517
3518
3519                                ;ERROR  273     PIP STIL SET AFTER SEARCH
3520  004502  073070                        EMT273
3521  004504  074104                        EHT1
3522  004506  074230                        EDT1
3523  004510  074320                        EFT1
3524
3525
3526                                ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
3527                                ;               REGISTERS BUT MXF DID NOT SET
3528  004512  073106                        EMT274
3529  004514  074104                        EHT1
3530  004516  074230                        EDT1
3531  004520  074320                        EFT1
3532
3533
3534                                ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
3535                                ;               COMMAND STARTED
3536  004522  073124                        EMT275
3537  004524  074104                        EHT1
3538  004526  074230                        EDT1
3539  004530  074320                        EFT1
3540
3541
3542                                ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
3543                                ;               ZERO
3544  004532  073136                        EMT276
3545  004534  074104                        EHT1
3546  004536  074230                        EDT1
3547  004540  074320                        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  073152                        EMT277
3554  004544  074104                        EHT1
3555  004546  074230                        EDT1
3556  004550  074320                        EFT1
3557
3558
3559                                ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
3560                                ;               WAS NOT VALID
3561  004552  073170                        EMT300
3562  004554  074104                        EHT1
3563  004556  074230                        EDT1
3564  004560  074320                        EFT1
3565
3566
3567                                ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
3568                                ;               IS VALID
3569  004562  073210                        EMT301
3570  004564  074104                        EHT1
3571  004566  074230                        EDT1
3572  004570  074320                        EFT1
3573
3574
3575                                ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
3576  004572  073232                        EMT302
3577  004574  074104                        EHT1
3578  004576  074230                        EDT1
3579  004600  074320                        EFT1
3580
3581
3582                                ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
3583  004602  073244                        EMT303
3584  004604  074104                        EHT1
3585  004606  074230                        EDT1
3586  004610  074320                        EFT1
3587
3588
3589                                ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
3590  004612  073256                        EMT304
3591  004614  074104                        EHT1
3592  004616  074230                        EDT1
3593  004620  074320                        EFT1
3594
3595
3596                                ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
3597  004622  073270                        EMT305
3598  004624  074104                        EHT1
3599  004626  074230                        EDT1
3600  004630  074320                        EFT1
3601
3602
3603                                ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
3604  004632  073302                        EMT306
3605  004634  074104                        EHT1
3606  004636  074230                        EDT1
3607  004640  074320                        EFT1
3608
```

```
3609
3610                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
3611  004642  073312                              EMT307
3612  004644  074104                              EHT1
3613  004646  074230                              EDT1
3614  004650  074320                              EFT1
3615
3616
3617                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
3618  004652  073332                              EMT310
3619  004654  074104                              EHT1
3620  004656  074230                              EDT1
3621  004660  074320                              EFT1
3622
3623
3624                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
3625  004662  073344                              EMT311
3626  004664  074104                              EHT1
3627  004666  074230                              EDT1
3628  004670  074320                              EFT1
3629
3630
3631                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
3632  004672  073356                              EMT312
3633  004674  074104                              EHT1
3634  004676  074230                              EDT1
3635  004700  074320                              EFT1
3636
3637
3638                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
3639  004702  073370                              EMT313
3640  004704  074104                              EHT1
3641  004706  074230                              EDT1
3642  004710  074320                              EFT1
3643
3644
3645                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
3646  004712  073410                              EMT314
3647  004714  074104                              EHT1
3648  004716  074230                              EDT1
3649  004720  074320                              EFT1
3650
3651
3652                                    ;ERROR  315     WRITE LOCK ERROR
3653  004722  073422                              EMT315
3654  004724  074104                              EHT1
3655  004726  074230                              EDT1
3656  004730  074320                              EFT1
3657
3658
3659                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
3660  004732  073434                              EMT316
3661  004734  074104                              EHT1
3662  004736  074230                              EDT1
3663  004740  074320                              EFT1
3664
```

```
3665
3666                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
3667   004742  073446                           EMT317
3668   004744  074104                           EHT1
3669   004746  074230                           EDT1
3670   004750  074320                           EFT1
3671
3672
3673                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
3674   004752  073456                           EMT320
3675   004754  074104                           EHT1
3676   004756  074230                           EDT1
3677   004760  074320                           EFT1
3678
3679
3680                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
3681   004762  073466                           EMT321
3682   004764  074104                           EHT1
3683   004766  074230                           EDT1
3684   004770  074320                           EFT1
3685
3686
3687                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
3688   004772  073476                           EMT322
3689   004774  074104                           EHT1
3690   004776  074230                           EDT1
3691   005000  074320                           EFT1
3692
3693
3694                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
3695   005002  073504                           EMT323
3696   005004  074104                           EHT1
3697   005006  074230                           EDT1
3698   005010  074320                           EFT1
3699
3700
3701                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3702   005012  073514                           EMT324
3703   005014  074104                           EHT1
3704   005016  074230                           EDT1
3705   005020  074320                           EFT1
3706
3707
3708                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3709   005022  073526                           EMT325
3710   005024  074104                           EHT1
3711   005026  074230                           EDT1
3712   005030  074320                           EFT1
3713
3714
3715                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
3716   005032  073540                           EMT326
3717   005034  074104                           EHT1
3718   005036  074230                           EDT1
3719   005040  074320                           EFT1
3720
```

```
3721
3722
3723  005042  073556                ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
3724  005044  074104                        EMT327
3725  005046  074230                        EHT1
3726  005050  074320                        EDT1
3727                                         EFT1
3728
3729
3730  005052  073570                ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3731  005054  074104                        EMT330
3732  005056  074230                        EHT1
3733  005060  074320                        EDT1
3734                                         EFT1
3735
3736
3737  005062  073600                ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3738  005064  074104                        EMT331
3739  005066  074230                        EHT1
3740  005070  074320                        EDT1
3741                                         EFT1
3742
3743
3744  005072  073612                ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3745  005074  074104                        EMT332
3746  005076  074230                        EHT1
3747  005100  074320                        EDT1
3748                                         EFT1
3749
3750
3751  005102  073624                ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3752  005104  074104                        EMT333
3753  005106  074230                        EHT1
3754  005110  074320                        EDT1
3755                                         EFT1
3756
3757
3758  005112  073642                ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3759  005114  074104                        EMT334
3760  005116  074230                        EHT1
3761  005120  074320                        EDT1
3762                                         EFT1
3763
3764
3765  005122  073660                ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3766  005124  074104                        EMT335
3767  005126  074230                        EHT1
3768  005130  074320                        EDT1
3769                                         EFT1
3770
3771
3772  005132  073700                ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3773  005134  074166                        EMT336
3774  005136  074264                        EHT336
3775  005140  074354                        EDT336
3776                                         EFT336
```

```
3777
3778                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3779  005142  073710                              EMT337
3780  005144  074200                              EHT337
3781  005146  074274                              EDT337
3782  005150  074364                              EFT337
3783
3784
3785                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3786  005152  073720                              EMT340
3787  005154  074166                              EHT336
3788  005156  074264                              EDT336
3789  005160  074354                              EFT336
3790
3791
3792                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3793  005162  073732                              EMT341
3794  005164  074166                              EHT336
3795  005166  074264                              EDT336
3796  005170  074354                              EFT336
3797
3798
3799                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3800  005172  073740                              EMT342
3801  005174  074104                              EHT1
3802  005176  074230                              EDT1
3803  005200  074320                              EFT1
3804
3805
3806                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3807  005202  073752                              EMT343
3808  005204  074104                              EHT1
3809  005206  074230                              EDT1
3810  005210  074320                              EFT1
3811
3812
3813                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3814  005212  073764                              EMT344
3815  005214  074212                              EHT344
3816  005216  074304                              EDT344
3817  005220  074374                              EFT344
3818
3819
3820                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3821  005222  073776                              EMT345
3822  005224  074104                              EHT1
3823  005226  074230                              EDT1
3824  005230  074320                              EFT1
3825
3826
3827                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3828  005232  074006                              EMT346
3829  005234  074104                              EHT1
3830  005236  074230                              EDT1
3831  005240  074320                              EFT1
3832
```

```
                                        ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
005242  074022                                  EMT347
005244  074104                                  EHT1
005246  074230                                  EDT1
005250  074320                                  EFT1

                                        ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
005252  074034                                  EMT350
005254  074104                                  EHT1
005256  074230                                  EDT1
005260  074320                                  EFT1

                                        ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
005262  074052                                  EMT351
005264  074104                                  EHT1
005266  074230                                  EDT1
005270  074320 .                                EFT1

                                        ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
005272  074062                                  EMT352
005274  000000                                  0
005276  000000                                  0
005300  000000                                  0

                                        ;ERROR  353     LOOK AHEAD TEST FAILS
005302  074066                                  EMT353
005304  074224                                  EHT353
005306  074316                                  EDT353
005310  074404                                  EFT353

                                        ;ERROR  354     BSE SHOULD NOT BE SET
005312  074076                                  EMT354
005314  074104                                  EHT1
005316  074230                                  EDT1
005320  074320                                  EFT1

                                        ;PUT ERROR TABLE HERE
                                        .SBTTL  ERROR TABLE USAGE

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

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

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005610  005227  177777                      INC     #-1             ;;FIRST TIME?
005614  001056                            BNE     69$             ;;BRANCH IF NO
005616  022737  042374  000042            CMP     #$ENDAD,@#42    ;;ACT-11?
005624  001452                            BEQ     69$             ;;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$:
005672  000427                            BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@CZRMCB - RM03/RM02 FUNCTIONAL TEST, PART 1 @<CRLF>
005752                                  69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005752  122737  000077  000041                CMPB    #77,@#41        ;RM03 IS A XXDP MEDIM ?
005760  001003                            BNE     5$              ;BRANCH IF NOT
005762  104401  067143                    TYPE    ,XDPMG          ;TYPE THE MESSAGE CHANGE PACK
005766  000000                            HALT                    ;HALT
005770                                  5$:
005770  005737  000042                    TST     42              ;IS LOC 42 ZERO ??
005774  001003                            BNE     10$             ;NO - NOT IN STANDALONE
005776  105737  001242                    TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
006002  001451                            BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
006004                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
006004  132737  000200  001243  XSIZ:    BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
006012  001043                            BNE     20$             ;NO
                                        ;       MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
006014  005037  001300                    CLR     $DEVM           ;CLEAR THE DEVICE MAP
006020  005001                            CLR     R1              ;START FROM DRIVE 0
006022  012704  000001                    MOV     #BIT0,R4        ;BIT MAP
006026  013700  001276                    MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
006032  012760  000040  000010  15$:      MOV     #CLR,RMCS2(R0)  ;MASS BUS CLEAR
006040  010160  000010                    MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
006044  016003  000010                    MOV     RMCS2(R0),R3    ;READ DRIVE STATUS TO SEEE NED BIT
006050  032703  010000                    BIT     #NED,R3         ;NED BIT SET ?
006054  001010                            BNE     16$             ;BRANCH IF SO
006056  016003  000000                    MOV     RMCS1(R0),R3    ;CHECK THE DVA BIT
006062  032703  004000                    BIT     #DVA,R3         ;DRIVE
006066  001403                            BEQ     16$             ;BRANCH IF DRIVE NOT AVAILABLE
006070  050437  001300                    BIS     R4,$DEVM                ;SET THE BIT MAP
006074  000405                            BR      17$             ;BRANCH NOT TYPE NONE EXIST MESG
006076                                  16$:
006076  104401  067230                    TYPE    ,NOTEX          ;MESSAGE NOT EXIST DRIVE
006102  010146                            MOV     R1,-(SP)        ;DRIVE NUMBER
```

```
 4020  006104  104403                        TYPOS
 4021  006106     006                        .BYTE   6
 4022  006107     000                        .BYTE   0
 4023  006110  005201              17$:      INC     R1              ;INCREMENT THE DRIVE ADDRESS
 4024  006112  006304                        ASL     R4              ;SET UP BIT MAP ADDRESS
 4025  006114  022701  000007                CMP     #7,R1           ;ALL DRIVES ARE CHECKED ?
 4026  006120  103344                        BHIS    15$             ;BRANCH IF NOT
 4027  006122                        20$:
 4028
 4029                                ;GO TO COMMON START CODE
 4030  006122  000137  006754                JMP     CMNSTART
```

```
4031  006126                          STANDALONE:
4032  006126  004737  064114                JSR     PC,$TKINT       ;INITIALIZE CONSOLE
4033
4034                                  ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
4035  006132  005327  000001                DEC     #1              ;FIRST START ??
4036  006136  100024                        BPL     10$             ;YES !!
4037
4038
4039                                  ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
4040  006140  104401  066452          5$:   TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
4041  006144  104411                        RDCHR                   ;GET RESPONSE
4042  006146  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
4043  006152  104401  001176                TYPE    ,$TMP1
4044  006156  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE??
4045  006164  001002                        BNE     6$              ;NO!!
4046  006166  000137  007106                JMP     READY           ;KEEP PREVIOUS PARAMETERS
4047  006172  123727  001176  000116  6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
4048  006200  001420                        BEQ     20$             ;GET NEW PARAMETERS
4049  006202  104401  066315                TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
4050  006206  000754                        BR      5$              ;RETRY
4051  006210                          10$:
4052
4053                                  ;SEE IF OPERATOR WANTS HELP FILE
4054  006210  104401  066317                TYPE    ,HELPQST        ;WANT HELP ??
4055  006214  104411                        RDCHR                   ;GET RESPONSE
4056  006216  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
4057  006222  104401  001176                TYPE    ,$TMP1
4058  006226  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
4059  006234  001002                        BNE     20$             ;NO - DONT TYPE HELP
4060  006236  104401  106126                TYPE    ,HELP           ;YES - TYPE HELP TEXT
4061  006242                          20$:
4062
4063                                  ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
4064  006242  104401  066353                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
4065  006246  104411                        RDCHR                   ;GET RESPONSE
4066  006250  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
4067  006254  104401  001176                TYPE    ,$TMP1
4068  006260  104411                        RDCHR                   ;WAIT ONE MORE CH, OR ANYTHING
4069  006262  005726                        TST     (SP)+           ;CLEAR THE STACK
4070  006264  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
4071  006272  001137                        BNE     90$             ;NO !!
4072  006274                          30$:
4073
4074                                  ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
4075  006274  104401  066511                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
4076  006300  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
4077  006304  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4078  006306  104401  066306                TYPE    ,CLSPRN
4079  006312  104413                        RDOCT                   ;GET NEW BUS ADDRESS
4080  006314  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
4081  006320  001412                        BEQ     50$             ;YES-SKIP TO NEXT ENTRY
4082  006322  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
4083  006330  101403                        BLOS    40$             ;YES
4084  006332  104401  066536                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
4085  006336  000756                        BR      30$             ;RETRY
4086  006340  013737  001176  001276  40$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
```

```
4087  006346  113737  001272  001176  50$:    MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
4088  006354  105037  001177                  CLRB    $TMP1+1
4089  006360  104401  066617                  TYPE    ,CNSL03
4090  006364  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
4091  006370  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
4092  006372     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
4093  006373     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
4094  006374  104401  066306                  TYPE    ,CLSPRN
4095  006400  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
4096  006402  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
4097  006406  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
4098  006410  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
4099  006416  101003                          BHI     60$             ;YES!!
4100  006420  104401  066647                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
4101  006424  000750                          BR      50$             ;RETRY
4102  006426  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
4103  006434  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
4104  006442  006237  001176                  ASR     $TMP1
4105  006446  006237  001176                  ASR     $TMP1
4106  006452  006237  001176                  ASR     $TMP1
4107  006456  006237  001176                  ASR     $TMP1
4108  006462  006237  001176                  ASR     $TMP1
4109  006466  105037  001177                  CLRB    $TMP1+1
4110  006472  104401  066723                  TYPE    ,CNSL05
4111  006476  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
4112  006502  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
4113  006504     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
4114  006505     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
4115  006506  104401  066306                  TYPE    ,CLSPRN
4116  006512  104413                          RDOCT                   ;GET NEW PRIORITY
4117  006514  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
4118  006520  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
4119  006522  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
4120  006530  002403                          BLT     80$             ;YES!!
4121  006532  104401  066757                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
4122  006536  000736                          BR      70$             ;RETRY
4123  006540                          80$:                            ;STORE NEW PRIORITY
4124  006540  006337  001176                  ASL     $TMP1
4125  006544  006337  001176                  ASL     $TMP1
4126  006550  006337  001176                  ASL     $TMP1
4127  006554  006337  001176                  ASL     $TMP1
4128  006560  006337  001176                  ASL     $TMP1
4129  006564  113737  001176  001273          MOVB    $TMP1,$VECT1+1
4130  006572                          90$:
4131
4132                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
4133  006572  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
4134  006576  104401  067004                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
4135  006602  104401  066311                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
4136  006606  104411                          RDCHR                   ;GET RESPONSE
4137  006610  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
4138  006614  104401  001176                  TYPE    ,$TMP1
4139  006620  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
4140  006626  001021                          BNE     110$            ;NO
4141  006630  000137  006004                  JMP     XSIZ            ;ALTO SIZE
4142  006634  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
```

```
4143  006642  000444                        BR      140$            ;SKIP TO NEXT ENTRY
4144  006644  104401  066311        100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
4145  006650  104411                        RDCHR                   ;GET RESPONSE
4146  006652  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
4147  006656  104401  001176                TYPE    ,$TMP1
4148  006662  023727  001176  000015        CMP     $TMP1,#CR       ;CARRIAGE RETURN??
4149  006670  001431                        BEQ     140$
4150  006672  023727  001176  000060  110$: CMP     $TMP1,#'0       ;NUMBER < 0??
4151  006700  002404                        BLT     120$            ;YES!!
4152  006702  023727  001176  000067        CMP     $TMP1,#'7       ;NUMBER > 7??
4153  006710  003403                        BLE     130$            ;NO!!
4154  006712  104401  066315        120$:   TYPE    ,QSTMRK         ;TYPE "?"
4155  006716  000752                        BR      100$            ;RETRY
4156  006720  013701  001176        130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
4157  006724  042701  177770                BIC     #^C7,R1
4158  006730  116102  067354                MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
4159  006734  042702  177400                BIC     #^C377,R2               ;CLEAR UNUSED BITS
4160  006740  050237  001300                BIS     R2,$DEVM                ;SET DEVICE # IN MAP
4161  006744  122737  000377  001300        CMPB    #377,$DEVM      ;DONE ??
4162  006752  101334                        BHI     100$            ;NO
4163  006754                        140$:
4164
```

```
 4165  006754                          CMNSTART:
 4166
 4167                                  ;ASSEMBLE TEST QUE FROM DEVICE MAP
 4168  006754  013700  001300                  MOV     $DEVM,R0          ;R0 = DEVICE MAP
 4169  006760  012701  001450                  MOV     #TSTQUE+2,R1      ;R1 = ADDRESS OF FIRST ENTRY IN QUE
 4170  006764  010137  001446                  MOV     R1,TSTQUE         ;INITIALIZE ENTRY POINTER
 4171  006770  012702  000001                  MOV     #1,R2             ;R2 = DEVICE POINTER
 4172  006774  005003                          CLR     R3                ;R3 = DEVICE NUMBER
 4173  006776  030200                  10$:    BIT     R2,R0             ;IS THIS DEVICE IN MAP ??
 4174  007000  001406                          BEQ     20$               ;NO !!
 4175  007002  010311                          MOV     R3,(R1)           ;YES - ENTER DEVICE NUMBER IN QUE
 4176  007004  116361  067354  000001          MOVB    ATNTBL(R3),1(R1)  ;ENTER ATTENTION BIT IN QUE
 4177  007012  062701  000002                  ADD     #2,R1             ;ADVANCE ENTRY POINTER
 4178  007016  006302                  20$:    ASL     R2                ;ADVANCE DEVICE POINTER
 4179  007020  105702                          TSTB    R2                ;DONE ALL DEVICES ??
 4180  007022  001402                          BEQ     25$               ;YES
 4181  007024  005203                          INC     R3                ;ADVANCE DEVICE NUMBER
 4182  007026  000763                          BR      10$               ;ENTER NEXT DEVICE
 4183  007030  005011                  25$:    CLR     (R1)              ;TERMINATE TEST QUE
 4184
 4185                                  ;SIZE FOR CLOCK
 4186  007032  004737  044704                  JSR     PC,SIZCLK         ;SEE IF CLOCK PRESENT
 4187  007036  000403                          BR      40$               ;YES - CLOCK IS PRESENT
 4188  007040  104000                  30$:    ERROR                     ;NO CLOCK
 4189  007042  000000                          HALT
 4190  007044  000775                          BR      30$
 4191  007046                          40$:
 4192                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 4193  007046  005737  000042                  TST     @#42              ;;ARE WE RUNNING UNDER XXDP/ACT?
 4194  007052  001012                          BNE     64$               ;;BRANCH IF YES
 4195  007054  123727  001242  000001          CMPB    $ENV,#1           ;;ARE WE RUNNING UNDER APT?
 4196  007062  001406                          BEQ     64$               ;;BRANCH IF YES
 4197  007064  023727  001154  000176          CMP     SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
 4198  007072  001005                          BNE     65$               ;;BRANCH IF NO
 4199  007074  104407                          GTSWR                     ;;GET SOFT-SWR SETTINGS
 4200  007076  000403                          BR      65$
 4201  007100  112737  000001  001150  64$:    MOVB    #1,$AUTOB         ;;SET AUTO-MODE INDICATOR
 4202  007106                          65$:
 4203  007106  000240                  READY:  NOP                       ;READY TO START TEST
 4204  007110  004737  064114                  JSR     PC,$TKINT         ;INITIALIZE TTY
 4205  007114  013746  000300                  MOV     PR6,-(SP)         ;;PUT NEW PS ON STACK
 4206  007120  012746  007126                  MOV     #64$,-(SP)        ;;PUT NEW PC ON  STACK
 4207  007124  000002                          RTI                       ;;POP NEW PC AND PS
 4208  007126                          64$:
 4209  007126  117737  172314  001234          MOVB    @TSTQUE,$UNIT     ;LOAD UNIT NUMBER
 4210  007134  005037  001472                  CLR     MEDENB            ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
007140                          TST1:
007140  000240                          NOP                             ;START OF TEST
007142  012737  007156  001122          MOV     #1$,$LPADR
007150  012737  007156  001124          MOV     #1$,$LPERR
007156                          1$:
007156  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007162  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007166  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007172  012737  000001  001226          MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
007200  005001                          CLR     R1
007202  013746  000004                  MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
007206  013746  000006                  MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
007212  012737  007304  000004          MOV     #3$,ERRVEC
007220  012737  000300  000006          MOV     #PR6,ERRVEC+2

007226  110160  000001                  MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
007232  010160  000002                  MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
007236  016002  000002                  MOV     RMWC(R0),R2
007242  010160  000004                  MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
007246  016002  000004                  MOV     RMBA(R0),R2
007252  010160  000010                  MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
007256  016002  000010                  MOV     RMCS2(R0),R2
007262  010160  000022                  MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
007266  016002  000022                  MOV     RMDB(R0),R2
007272  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007276  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007302  000415                          BR      7$                      ;NO BUS TIMEOUT OCCURRED

007304  022626                  3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
007306  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007312  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007316  104110                          ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
007320  005737  000042                  TST     42                      ;STAND ALONE MODE??
007324  001002                          BNE     5$                      ;NO!!
007326  000137  005322                  JMP     START                   ;YES-GO GET $BASE
007332  000137  042204          5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
007336                          7$:

                                ;;*****************************************************************
                                ;*TEST 2         DEVICE AVAILABLE TEST
                                ;;*****************************************************************
007336                          TST2:
007336  000004                          SCOPE                           ;SCOPE CALL
007340  000240                          NOP                             ;START OF TEST
007342  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007346  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007352  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007356  012737  000002  001226          MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

007364  004737  053360                  JSR     PC,CNTCLR
007370  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007372  000240                          NOP                             ;RETURN HERE IF ERROR
```

```
                                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
                                        JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
                                2$:
                                        MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
                                        MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
                                        MOV     #5$,ERRVEC
                                        MOV     PR6,ERRVEC+2

                                        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
                                        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
                                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
                                        BEQ     3$              ;NO!!
                                        ERROR   111             ;NONEXISTENT DEVICE
                                        BR      7$
                                3$:     BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
                                        BNE     9$              ;YES!!
                                        ERROR   112             ;DEVICE NOT AVAILABLE
                                        BR      7$

                                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
                                7$:     JMP     $EOSP

                                9$:

                                ;;*****************************************************************
                                ;*TEST 3         DRIVE TYPE TEST
                                ;;*****************************************************************
                                TST3:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #3,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,CNTCLR
                                        BR      2$      ;GO TO 2$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
                                        JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
                                2$:
                                        MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
                                        MOVB    #200,GETINX+1
                                        MOV     #5$,RMDTI               ;RMDT INPUT BUFFER = 5$
                                        JSR     PC,GET                  ;GO GET DRIVE TYPE
                                        BR      3$      ;GO TO 3$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR NUMBER SPECIFIED BY GET
                                3$:     CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
                                        BEQ     5$              ;YES!!
```

```
4323  007632  022737  024024  001354          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
4324  007640  001421                          BEQ     5$              ;YES!!
4325  007642  022737  020025  001354          CMP     #SNGPRT!BIT0,RMDTI      ;SINGLE PROT RM02 ?
4326  007650  001415                          BEQ     5$              ;YES
4327  007652  022737  024025  001354          CMP     #DULPRT!BIT0,RMDTI      ;DUAL PORT RM02 ?
4328  007660  001411                          BEQ     5$              ;YES
4329  007662  012737  020024  001176          MOV     #SNGPRT,$TMP1
4330  007670  012737  024024  001200          MOV     #DULPRT,$TMP2
4331  007676  104114                          ERROR   114             ;NOT AN RM03
4332  007700  000137  042150          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
4333
4334  007704                          5$:
4335                                  ;;*****************************************************************
4336                                  ;*TEST 4         UNIBUS INITIALIZE TEST
4337
4338                                  ;;*****************************************************************
4339  007704                          TST4:
4340  007704  000004                          SCOPE                   ;SCOPE CALL
4341  007706  000240                          NOP                     ;START OF TEST
4342  007710  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
4343  007716  012737  007732  001122          MOV     #1$,$LPADR
4344  007724  012737  007732  001124          MOV     #1$,$LPERR
4345  007732                          1$:
4346  007732  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4347  007736  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4348  007742  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4349  007746  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4350
4351
4352  007754  004737  053360                  JSR     PC,CNTCLR
4353  007760  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4354  007762  000240                          NOP             ;RETURN HERE IF ERROR
4355  007764  104000                          ERROR           ;ERROR NUMBER DEFINED BY SUB
4356  007766  000137  010570                  JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4357  007772  012702  000101          10$:    MOV     #65.,R2         ;SET OR AND RESET IR
4358  007776  012737  000000  001420          MOV     #0,RMDB0        ;WRITE ZEROS IN DATA SILO
4359  010004  112737  000022  001533          MOVB    #RMDB,PUTINX    ;SETUP REGISTER INDEX
4360  010012  112737  000200  001534          MOVB    #200,PUTINX+1
4361  010020                          20$:
4362  010020  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4363  010024  000404                          BR      30$     ;GO TO 30$ IF NO ERROR
4364  010026  000240                          NOP             ;RETURN HERE IF ERROR
4365  010030  104000                          ERROR           ;ERROR DEFINED BY PUT SUB
4366  010032  000137  010570                  JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4367  010036  005302                  30$:    DEC     R2              ;DECREMENT COUNT
4368  010040  001367                          BNE     20$             ;WRITE SILO AGAIN
4369  010042  012737  177777  001402          MOV     #-1,RMBA0       ;RMBA=ALL ONES
4370  010050  012737  177777  001412          MOV     #-1,RMER10      ;RMER1=ALL ONES
4371  010056  012737  177777  001440          MOV     #-1,RMER20      ;RMER2 = ALL ONES
4372  010064  012737  040001  001422          MOV     #DMD!DBEN,RMMR10        ;SET DIAGNOSTIC MODE
4373  010072  012737  003577  001376          MOV     #003577,RMCS10
4374  010100  012737  021037  001406          MOV     #021037,RMCS20
4375
4376  010106  012702  001533                  MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
4377  010112  112722  000004                  MOVB    #RMBA,(R2)+
4378  010116  112722  000014                  MOVB    #RMER1,(R2)+
```

```
4379  010122  112722  000042                        MOVB    #RMER2,(R2)+
4380  010126  112722  000024                        MOVB    #RMMR1,(R2)+
4381  010132  112722  000000                        MOVB    #RMCS1,(R2)+
4382  010136  112722  000010                        MOVB    #RMCS2,(R2)+
4383  010142  112722  000200                        MOVB    #200,(R2)+
4384  010146  004737  044466                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4385  010152  000404                                BR      40$     ;GO TO 40$ IF NO ERROR
4386  010154  000240                                NOP                             ;RETURN HERE IF ERROR
4387  010156  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
4388  010160  000137  010570                        JMP     220$    ;GO TO 220$ IF ERROR WAS FOUND
4389  010164  000005                        40$:    RESET                           ;UNIBUS INITIALIZE
4390  010166  004737  064114                        JSR     PC,$TKINT               ;INITIALIZE CONSOLE
4391  010172  111160  000010                        MOVB    (R1),RMCS2(R0)          ;SELECT DEVICE
4392  010176  004737  044132                        JSR     PC,GETSTS               ;GO SET UP FOR STATUS FETCH
4393  010202  004737  044216                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4394  010206  000403                                BR      50$     ;GO TO 50$ IF NO ERROR
4395  010210  000240                                NOP                             ;RETURN HERE IF ERROR
4396  010212  104000                                ERROR                           ;ERROR DEFINED BY GET SUB
4397  010214  000565                                BR      220$                    ;SKIP REMAINDER OF TEST
4398
4399  010216  013737  001326  001142  50$:   MOV     RMCS1I,$BDDAT           ;VERIFY RMCS1
4400  010224  042737  100000  001142        BIC     #SC,$BDDAT              ;IGNORE SPECIAL CONDITION
4401  010232  012737  004200  001140        MOV     #DVA!RDY,$GDDAT         ;EXPECT DVA & RDY
4402  010240  023737  001140  001142        CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED, RECEIVED
4403  010246  001401                        BEQ     60$                     ;BRANCH IF EQUAL
4404  010250  104115                        ERROR   115                     ;RMCS1 NOT INITIALIZED
4405
4406  010252  005037  001140        60$:   CLR     $GDDAT                  ;VERIFY RMBA IS ZERO
4407  010256  013737  001332  001142        MOV     RMBAI,$BDDAT
4408  010264  001401                        BEQ     70$                     ;BRANCH IF ZERO
4409  010266  104116                        ERROR   116                     ;RMBA NOT INITIALIZED
4410
4411  010270  013737  001336  001142  70$:   MOV     RMCS2I,$BDDAT           ;VERIFY RMCS2
4412  010276  005046                        CLR     -(SP)                   ;EXPECT IR & UNIT NUMBER
4413  010300  111116                        MOVB    (R1),(SP)
4414  010302  052716  000100                BIS     #IR,(SP)
4415  010306  012637  001140                MOV     (SP)+,$GDDAT
4416  010312  023737  001140  001142        CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED & RECEIVED
4417  010320  001401                        BEQ     90$                     ;BRANCH IF EQUAL
4418  010322  104117                        ERROR   117                     ;RMCS2 NOT INITIALIZED
4419
4420  010324  005037  001140        90$:   CLR     $GDDAT                  ;VERIFY RMER1
4421  010330  013737  001342  001142        MOV     RMER1I,$BDDAT
4422  010336  042737  040000  001142        BIC     #UNS,$BDDAT             ;IGNORE UNSAFE
4423  010344  001401                        BEQ     110$                    ;BRANCH IF ZERO
4424  010346  104120                        ERROR   120                     ;RMER1 NOT INITIALIZED
4425
4426  010350  013737  001344  001142  110$:   MOV     RMAS1,$BDDAT            ;VERIFY RMAS
4427  010356  005002                        CLR     R2                      ;CLEAR ALL BUT THIS
4428  010360  116102  000001                MOVB    1(R1),R2                ;DRIVES ATTENTION BIT
4429  010364  000302                        SWAB    R2
4430  010366  005102                        COM     R2
4431  010370  000240                        NOP
4432  010372  040237  001142                BIC     R2,$BDDAT
4433  010376  001401                        BEQ     130$                    ;BRANCH IF ITS 0
4434  010400  104121                        ERROR   121                     ;RMAS NOT INITIALIZED
```

```
4435
4436  010402  013737  001352  001142  130$:   MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
4437  010410  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
4438  010416  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
4439  010424  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
4440  010432  001401                          BEQ     170$            ;BRANCH IF 0
4441  010434  104122                          ERROR   122             ;RMMR NOT INITIALIZED
4442
4443  010436  005037  001140          170$:   CLR     $GDDAT          ;EXPECT ZEROS
4444  010442  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
4445  010450  001401                          BEQ     190$
4446  010452  104124                          ERROR   124             ;RMEC2 NOT INITIALIZED
4447
4448  010454  013737  001366  001142  190$:   MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
4449  010462  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
4450  010470  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST,TAG BIT ON
4451  010476  023737  001140  001142          CMP     $GDDAT,$BDDAT
4452  010504  001401                          BEQ     210$
4453  010506  104125                          ERROR   125             ;RMER2 NOT INITIALIZED
4454
4455  010510  005037  001140          210$:   CLR     $GDDAT          ;EXPECT ALL ZEROS
4456  010514  013737  001370  001142          MOV     RMER2I,$BDDAT   ;VERIFY RMER2
4457  010522  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
4458  010530  001401                          BEQ     215$            ;BRANCH IF OTHER BITS 0
4459  010532  104173                          ERROR   173
4460  010534  013737  001340  001142  215$:   MOV     RMDSI,$BDDAT    ;CHECK DRIVE STATUS REGISTER
4461  010542  042737  177177  001142          BIC     #^C<DPR!DRY>,$BDDAT
4462  010550  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED STATUS
4463  010556  023737  001142  001140          CMP     $BDDAT,$GDDAT   ;COMPARE EXPECTED & RECEIVED STATUS
4464  010564  001401                          BEQ     220$            ;DRIVE STATUS IS OK
4465  010566  104123                          ERROR   123             ;REPORT ERROR IN DRIVE STATUS
4466  010570                          220$:
4467
4468
4469                                  ;;*****************************************************************
4470                                  ;*TEST 5         CONTROLLER CLEAR TEST
4471
4472                                  ;;*****************************************************************
4473  010570                          TST5:
4474  010570  000004                          SCOPE                   ;SCOPE CALL
4475  010572  000240                          NOP                     ;START OF TEST
4476  010574  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4477  010600  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4478  010604  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4479  010610  012737  000005  001226          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4480
4481  010616  004737  053360                  JSR     PC,CNTCLR
4482  010622  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4483  010624  000240                          NOP                     ;RETURN HERE IF ERROR
4484  010626  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4485  010630  000137  011126                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4486  010634  012702  000101          10$:    MOV     #65.,R2
4487  010640  012737  000000  001420          MOV     #0,RMDB0                ;WRITE ZEROS IN DATA SILO
4488  010646  112737  000022  001533          MOVB    #RMDB,PUTINX    ;SETUP REGISTER INDEX TABLE
4489  010654  112737  000200  001534          MOVB    #200,PUTINX+1
4490  010662                          20$:
```

```
4491  010662  004737  044466                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4492  010666  000404                            BR      30$     ;GO TO 30$ IF NO ERROR
4493  010670  000240                            NOP                     ;RETURN HERE IF ERROR
4494  010672  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
4495  010674  000137  011126                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4496  010700  005302                    30$:    DEC     R2              ;DECREMENT COUNT
4497  010702  001367                            BNE     20$             ;AND WRITE SILO AGAIN IF NOT DONE
4498  010704  012737  177777  001402            MOV     #-1,RMBA0
4499  010712  012737  177777  001412            MOV     #-1,RMER10
4500  010720  012737  177777  001440            MOV     #-1,RMER20
4501  010726  012737  040001  001422            MOV     #DMD!DBEN,RMMR10
4502  010734  012737  003577  001376            MOV     #003577,RMCS10
4503  010742  012737  021037  001406            MOV     #021037,RMCS20
4504  010750  012702  001533                    MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
4505  010754  112722  000004                    MOVB    #RMBA,(R2)+
4506  010760  112722  000014                    MOVB    #RMER1,(R2)+
4507  010764  112722  000042                    MOVB    #RMER2,(R2)+
4508  010770  112722  000024                    MOVB    #RMMR1,(R2)+
4509  010774  112722  000000                    MOVB    #RMCS1,(R2)+
4510  011000  112722  000010                    MOVB    #RMCS2,(R2)+
4511  011004  112722  000200                    MOVB    #200,(R2)+
4512  011010  004737  044466                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4513  011014  000403                            BR      40$     ;GO TO 40$ IF NO ERROR
4514  011016  000240                            NOP                     ;RETURN HERE IF ERROR
4515  011020  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
4516  011022  000430                            BR      70$
4517
4518  011024                            40$:
4519  011024  004737  053360                    JSR     PC,CNTCLR
4520  011030  000404                            BR      50$     ;GO TO 50$ IF NO ERROR
4521  011032  000240                            NOP                     ;RETURN HERE IF ERROR
4522  011034  104000                            ERROR                   ;ERROR NUMBER DEFINED BY SUB
4523  011036  000137  011126                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4524  011042  004737  044132            50$:    JSR     PC,GETSTS
4525  011046  004737  044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4526  011052  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
4527  011054  000240                            NOP                     ;RETURN HERE IF ERROR
4528  011056  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
4529  011060  000137  011126                    JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
4530  011064                            60$:
4531  011064  004737  053476                    JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
4532  011070  000405                            BR      70$             ;GO TO 70$ IF NO ERROR
4533  011072  000240                            NOP                     ;RETURN HERE IF ERROR
4534  011074  104000                            ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
4535  011076  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4536  011100  000137  011126                    JMP     80$             ;GO TO 80$ IF ERROR WAS FOUND
4537  011104                            70$:
4538  011104  012737  000000  001376            MOV     #NOP,RMCS10     ;CHANGE  FUNCTION CODE FOR ERROR CHECK
4539  011112  004737  046044                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4540  011116  000403                            BR      80$             ;GO TO 80$ IF NO ERROR
4541  011120  000240                            NOP                     ;RETURN HERE IF ERROR
4542  011122  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4543  011124  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4544  011126                            80$:
4545
4546                                    ;;*****************************************************************
```

```
                                        ;*TEST 6        ERROR CLEAR TEST

                                        ;;*****************************************************************
011126                                  TST6:
                                                SCOPE                   ;SCOPE CALL
011126  000004                                  NOP                     ;START OF TEST
011130  000240                                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
011132  012706  001100                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
011136  013700  001276                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
011142  013701  001446                          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
011146  012737  000006  001226
                                                JSR     PC,CNTCLR
011154  004737  053360                          BR      10$     ;GO TO 10$ IF NO ERROR
011160  000404                                  NOP                     ;RETURN HERE IF ERROR
011162  000240                                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
011164  104000                                  JMP     50$     ;GO TO 50$ IF ERROR WAS FOUND
011166  000137  011332                  10$:
011172                                          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
011172  112737  000000  001533                  MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
011200  112737  000200  001534                  MOV     #TRE,RMCS10     ;RMCS1 OUTPUT BUFFER = TRE
011206  012737  040000  001376                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
011214  004737  044466                          BR      20$     ;GO TO 20$ IF NO ERROR
011220  000403                                  NOP                     ;RETURN HERE TO REPORT AN ERROR
011222  000240                                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
011224  104000                                  BR      50$             ;SKIP REST OF TEST
011226  000441
011230  112737  000000  001504  20$:           MOVB    #RMCS1,GETINX
011236  112737  000010  001505                  MOVB    #RMCS2,GETINX+1
011244  112737  000200  001506                  MOVB    #200,GETINX+2
011252  004737  044216                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
011256  000403                                  BR      30$     ;GO TO 30$ IF NO ERROR
011260  000240                                  NOP                     ;RETURN HERE IF ERROR
011262  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
011264  000422                                  BR      50$             ;SKIP REST OF TEST
011266  013737  001326  001142  30$:           MOV     RMCS1I,$BDDAT   ;CHECK TRE & MCPE
011274  005037  001140                          CLR     $GDDAT          ;EXPECT 0'S
011300  042737  117777  001142                  BIC     #^C<TRE!MCPE>,$BDDAT
011306  001401                                  BEQ     40$             ;BRANCH IF TRE & MCPE = 0
011310  104137                                  ERROR   137             ;MCPE, OR TRE NOT CLEARED

011312  013737  001336  001142  40$:           MOV     RMCS2I,$BDDAT   ;CHECK RMCS2
011320  042737  104377  001142                  BIC     #^C<WCE!UPE!NED!PGE!MXF!MDPE>,$BDDAT
011326  001401                                  BEQ     50$
011330  104140                                  ERROR   140             ;RMCS2 NOT CLEARED

011332                                  50$:

                                        ;;*****************************************************************
                                        ;*TEST 7        DRIVE STATUS TEST

                                        ;;*****************************************************************
011332                                  TST7:
                                                SCOPE                   ;SCOPE CALL
011332  000004                                  NOP                     ;START OF TEST
011334  000240                                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
011336  012706  001100                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
011342  013700  001276                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
011346  013701  001446
```

```
4603  011352  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4604
4605  011360  004737  053360                  JSR     PC,CNTCLR
4606  011364  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
4607  011366  000240                          NOP                     ;RETURN HERE IF ERROR
4608  011370  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4609  011372  000137  011656                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
4610  011376                          10$:
4611  011376  004737  044132                  JSR     PC,GETSTS
4612  011402  004737  044216                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4613  011406  000403                          BR      20$     ;GO TO 20$ IF NO ERROR
4614  011410  000240                          NOP                     ;RETURN HERE IF ERROR
4615  011412  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4616  011414  000520                          BR      100$            ;SKIP REST OF TEST
4617  011416  032737  010000  001340  20$:    BIT     #MOL,RMDSI      ;MEDIUM ON LINE??
4618  011424  001016                          BNE     30$             ;YES!!
4619  011426  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
4620  011434  042737  162000  001140          BIC     #ATA!ERR!PIP!LBT,$GDDAT
4621  011442  052737  010000  001140          BIS     #MOL,$GDDAT
4622  011450  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
4623  011456  104151                          ERROR   151             ;MOL STATUS INCORRECT
4624  011460  000462                          BR      80$
4625
4626  011462  032737  000200  001370  30$:    BIT     #DVC,RMER2I     ;IS THERE A DEVICE CHECK??
4627  011470  001406                          BEQ     50$
4628  011472  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
4629  011500  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
4630  011504  104152                          ERROR   152             ;DRIVE FAULT
4631
4632  011506  032737  000200  001370  50$:    BIT     #DVC,RMER2I     ;WAS DVC SET??
4633  011514  001414                          BEQ     60$             ;NO!!
4634  011516  032737  040000  001342          BIT     #UNS,RMER1I     ;IS UNS SET??
4635  011524  001022                          BNE     70$             ;YES!!
4636  011526  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
4637  011534  012737  040000  001140          MOV     #UNS,$GDDAT     ;EXPECTED STATUS
4638  011542  104153                          ERROR   153             ;DVC IS SET BUT UNS IS NOT
4639  011544  000412                          BR      70$
4640  011546  032737  040000  001342  60$:    BIT     #UNS,RMER1I     ;IS UNS SET??
4641  011554  001406                          BEQ     70$             ;NO!!
4642  011556  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
4643  011564  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
4644  011570  104154                          ERROR   154             ;DVC IS NOT SET BUT UNS IS
4645  011572  032737  004000  001340  70$:    BIT     #WRL,RMDSI      ;IS WRITE PROTECT ON??
4646  011600  001414                          BEQ     90$             ;NO!!
4647  011602  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
4648  011610  042737  166076  001140          BIC     #^C<MOL!PGM!DPR!DRY!VV!OM>,$GDDAT
4649  011616  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
4650  011624  104175                          ERROR   175             ;SELECTED DEVICE IN WRITE PROTECT
4651  011626  000137  042150          80$:    JMP     $EOSP   ;SKIP REST OF TEST
4652
4653  011632  032737  040000  001370  90$:    BIT     #SKI,RMER2I     ;IS SKI SET??
4654  011640  001406                          BEQ     100$            ;NO!!
4655  011642  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
4656  011650  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
4657  011654  104205                          ERROR   205             ;PERSISTENT SEEK INCOMPLETE ERROR
4658  011656                          100$:
```

```
                                ;;*****************************************************************
                                ;*TEST 10        PRIMARY/SECONDARY ERROR TEST
                                ;*****************************************************************
011656                          TST10:
011656  000004                          SCOPE                   ;SCOPE CALL
011660  000240                          NOP                     ;START OF TEST
011662  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
011666  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
011672  013701  001446                  MOV     TSTGUE,R1       ;(R1) = DEVICE BEING TESTED
011676  012737  000010  001226          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                        
011704  004737  053360                  JSR     PC,CNTCLR
011710  000404                          BR      1$      ;GO TO 1$ IF NO ERROR
011712  000240                          NOP                     ;RETURN HERE IF ERROR
011714  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
011716  000137  012036                  JMP     5$      ;GO TO 5$ IF ERROR WAS FOUND
011722                          1$:
011722  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
011730  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
011736  012737  000023  001376          MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
011744  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
011750  000403                          BR      2$      ;GO TO 2$ IF NO ERROR
011752  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
011754  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
011756  000427                          BR      5$              ;SKIP REST OF TEST IF ERROR
011760  004737  044132          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
011764  004737  045026                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION OF NOP
011770  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
011774  000403                          BR      3$      ;GO TO 3$ IF NO ERROR
011776  000240                          NOP                     ;RETURN HERE IF ERROR
012000  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
012002  000415                          BR      5$      ;SKIP REST OF TEST IF ERROR
012004                          3$:
012004  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
012010  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
012012  000240                          NOP                     ;RETURN HERE IF ERROR
012014  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
012016  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012020  000406                          BR      5$              ;SKIP REST OF TEST IF ERROR
012022                          4$:
012022  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
012026  000403                          BR      5$              ;GO TO 5$ IF NO ERROR
012030  000240                          NOP                     ;RETURN HERE IF ERROR
012032  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
012034  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012036                          5$:
                        
                                ;;*****************************************************************
                                ;*TEST 11        DIAGNOSTIC MODE TEST
                                ;*****************************************************************
012036                          TST11:
012036  000004                          SCOPE                   ;SCOPE CALL
012040  000240                          NOP                     ;START OF TEST
012042  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
012046  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
```

```
 4715  012052  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 4716  012056  012737  000011  001226            MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 4717
 4718  012064  004737  053360                    JSR     PC,CNTCLR
 4719  012070  000404                            BR      1$      ;GO TO 1$ IF NO ERROR
 4720  012072  000240                            NOP                     ;RETURN HERE IF ERROR
 4721  012074  104000                            ERROR                   ;ERROR NUMBER DEFINED BY SUB
 4722  012076  000137  013110                    JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
 4723  012102                          1$:
 4724  012102  112737  000024  001533            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
 4725  012110  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
 4726  012116  012737  000001  001422            MOV     #DMD,RMMR10     ;RMMR1=DMD
 4727  012124  004737  044466                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
 4728  012130  000404                            BR      2$              ;GO TO 2$ IF NO ERROR
 4729  012132  000240                            NOP                     ;RETURN HERE IF ERROR
 4730  012134  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
 4731  012136  000137  013110                    JMP     22$             ;GO TO 22$ IF ERROR WAS FOUND
 4732  012142  004737  044132          2$:       JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
 4733  012146  004737  044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4734  012152  000404                            BR      3$      ;GO TO 3$ IF NO ERROR
 4735  012154  000240                            NOP                     ;RETURN HERE IF ERROR
 4736  012156  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
 4737  012160  000137  013110                    JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
 4738  012164  032737  000001  001352  3$:       BIT     #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
 4739  012172  001011                            BNE     5$              ;YES!!
 4740  012174  012737  000001  001140            MOV     #DMD,$GDDAT     ;EXPECTED STATUS
 4741  012202  013737  001352  001142            MOV     RMMR1I,$BDDAT   ;RECEIVED STATUS
 4742  012210  104176                            ERROR   176             ;COULD NOT SET DIAGNOSTIC MODE
 4743  012212  000137  013110                    JMP     22$             ;SKIP REST OF  TEST IF DMD = 0
 4744  012216  032737  010000  001340  5$:       BIT     #MOL,RMDSI      ;IS "MOL" = 0   ??
 4745  012224  001411                            BEQ     6$              ;YES!!
 4746  012226  013737  001340  001142            MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4747  012234  042737  177777  001142            BIC     #^CMOL,$BDDAT
 4748  012242  005037  001140                    CLR     $GDDAT          ;SETUP GOOD DATA FOR TYPEOUT
 4749  012246  104177                            ERROR   177             ;INCORRECT MOL STATUS
 4750  012250  032737  020000  001340  6$:       BIT     #PIP,RMDSI      ;IS PIP  SET??
 4751  012256  001012                            BNE     7$              ;YES!!
 4752  012260  013737  001340  001142            MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4753  012266  042737  157777  001142            BIC     #^CPIP,$BDDAT
 4754  012274  012737  020000  001140            MOV     #PIP,$GDDAT     ;EXPECTED PIP SET
 4755  012302  104200                            ERROR   200             ;INCORRECT PIP STATUS
 4756  012304  032737  004000  001340  7$:       BIT     #WRL,RMDSI      ;IS WRITE LOCK OFF??
 4757  012312  001411                            BEQ     8$              ;YES!!
 4758  012314  013737  001340  001142            MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
 4759  012322  042737  173777  001142            BIC     #^CWRL,$BDDAT
 4760  012330  005037  001140                    CLR     $GDDAT          ;EXPECTED WRL  = 0
 4761  012334  104201                            ERROR   201             ;INCORRECT WRL STATUS
 4762  012336  032737  040000  001370  8$:       BIT     #SKI,RMER2I     ;IS SKI = 0
 4763  012344  001411                            BEQ     9$              ;YES!!
 4764  012346  013737  001370  001142            MOV     RMER2I,$BDDAT   ;SETUP BAD DATA FOR TYPEOUT
 4765  012354  042737  137777  001142            BIC     #^CSKI,$BDDAT
 4766  012362  005037  001140                    CLR     $GDDAT          ;SKI SHOULD BE 0
 4767  012366  104202                            ERROR   202             ;REPORT BAD SKI STATUS
 4768  012370  032737  000200  001370  9$:       BIT     #DVC,RMER2I     ;IS DEVICE CHECK  = 0??
 4769  012376  001411                            BEQ     10$             ;YES!!
 4770  012400  013737  001370  001142            MOV     RMER2I,$BDDAT   ;SETUP BAD DATA FOR  TYPEOUT
```

```
4771  012406  042737  177577  001142          BIC     #↑CDVC,$BDDAT
4772  012414  005037  001140                  CLR     $GDDAT          ;DVC SHOULD BE 0
4773  012420  104203                          ERROR   203             ;REPORT BAD DVC STATUS
4774  012422                          10$:
4775  012422  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4776  012430  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4777  012436  012737  001711  001422          MOV     #DMD!MUR!MOC!MSER!MDF!MWP,RMMR1O        ;RMMR1=DMD!MUR!MOC!MSER!MDF!MWP
4778  012444  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4779  012450  000404                          BR      11$             ;GO TO 11$ IF NO ERROR
4780  012452  000240                          NOP                     ;RETURN HERE IF ERROR
4781  012454  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4782  012456  000137  013110                  JMP     22$             ;GO TO 22$ IF ERROR WAS FOUND
4783  012462                          11$:
4784  012462  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4785  012466  000404                          BR      12$     ;GO TO 12$ IF NO ERROR
4786  012470  000240                          NOP                     ;RETURN HERE IF ERROR
4787  012472  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4788  012474  000137  013110                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4789  012500  032737  000001  001352  12$:     BIT     #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
4790  012506  001011                          BNE     125$            ;YES!!
4791  012510  012737  000001  001140          MOV     #DMD,$GDDAT     ;EXPECTED STATUS
4792  012516  013737  001352  001142          MOV     RMMR1I,$BDDAT   ;RECEIVED STATUS
4793  012524  104176                          ERROR   176             ;COULD NOT SET DIAGNOSTIC MODE
4794  012526  000137  013110                  JMP     22$             ;SKIP REST OF TEST IF DMD = 0
4795  012532  032737  010000  001340  125$:    BIT     #MOL,RMDSI      ;IS MOL = 1??
4796  012540  001012                          BNE     13$             ;YES!!
4797  012542  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4798  012550  042737  167777  001142          BIC     #↑CMOL,$BDDAT
4799  012556  012737  010000  001140          MOV     #MOL,$GDDAT     ;EXPECTED MOL  =1
4800  012564  104177                          ERROR   177             ;REPORT MOL STATUS ERROR
4801  012566  032737  020000  001340  13$:     BIT     #PIP,RMDSI      ;IS   PIP 0 ??
4802  012574  001411                          BEQ     14$             ;YES!!
4803  012576  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4804  012604  042737  157777  001142          BIC     #↑CPIP,$BDDAT
4805  012612  005037  001140                  CLR     $GDDAT          ;EXPECTED PIP STATUS
4806  012616  104200                          ERROR   200             ;REPORT BAD PIP STATUS
4807  012620  032737  004000  001340  14$:     BIT     #WRL,RMDSI      ;IS WRL SET??
4808  012626  001012                          BNE     15$             ;YES!!
4809  012630  013737  001340  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
4810  012636  042737  173777  001142          BIC     #↑CWRL,$BDDAT
4811  012644  012737  004000  001140          MOV     #WRL,$GDDAT     ;EXPECTED GOOD STATUS
4812  012652  104201                          ERROR   201             ;REPORT ERROR IN  WRL STATUS
4813  012654  032737  040000  001370  15$:     BIT     #SKI,RMER2I     ;IS SKI SET??
4814  012662  001012                          BNE     16$             ;YES!!
4815  012664  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
4816  012672  042737  137777  001142          BIC     #↑CSKI,$BDDAT
4817  012700  012737  040000  001140          MOV     #SKI,$GDDAT     ;EXPECTED SKI  ON
4818  012706  104202                          ERROR   202
4819  012710  032737  000200  001370  16$:     BIT     #DVC,RMER2I     ;IS DVC SET  ??
4820  012716  001012                          BNE     17$             ;YES!!
4821  012720  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
4822  012726  042737  177577  001142          BIC     #↑CDVC,$BDDAT
4823  012734  012737  000200  001140          MOV     #DVC,$GDDAT     ;EXPECTED DVC ON
4824  012742  104203                          ERROR   203             ;REPORT DVC STATUS ERROR
4825  012744  032737  000100  001340  17$:     BIT     #VV,RMDSI       ;MUR SHOULD HAVE RESET VOLUME VALID
4826  012752  001411                          BEQ     19$             ;BRANCH IF IT DID
```

```
4827  012754  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
4828  012762  042737  177677  001142          BIC     #↑CVV,$BDDAT
4829  012770  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
4830  012774  104204                          ERROR   204
4831  012776                          19$:
4832  012776  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4833  013004  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4834  013012  012737  000011  001376          MOV     #DRVCLR!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = DRVCLR!GO
4835  013020  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
4836  013024  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
4837  013026  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
4838  013030  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4839  013032  000137  013110                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4840  013036                          20$:
4841  013036  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4842  013042  000404                          BR      21$     ;GO TO 21$ IF NO ERROR
4843  013044  000240                          NOP                     ;RETURN HERE IF ERROR
4844  013046  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4845  013050  000137  013110                  JMP     22$     ;GO TO 22$ IF ERROR WAS FOUND
4846  013054  032737  002000  001370  21$:    BIT #LBC,RMER2I         ;IS LBC SET ??
4847  013062  001012                          BNE 22$                 ;YES!!
4848  013064  013737  001370  001142          MOV RMER2I,$BDDAT       ;BAD DATA FOR TYPEOUT
4849  013072  012737  002000  001140          MOV #LBC, $GDDAT        ;GOOD DATA FOR TYPEOUT
4850  013100  042737  002000  001142          BIC #LBC, $BDDAT
4851  013106  104262                          ERROR   262
4852  013110                          22$:
4853                                  ;;*****************************************************************
4854                                  ;*TEST 12        PACK ACKNOWLEDGE TEST
4855
4856                                  ;;*****************************************************************
4857  013110                          TST12:
4858  013110  000004                          SCOPE                   ;SCOPE CALL
4859  013112  000240                          NOP                     ;START OF TEST
4860  013114  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4861  013120  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4862  013124  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4863  013130  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4864
4865  013136  004737  053360                  JSR     PC,CNTCLR
4866  013142  000404                          BR      1$      ;GO TO 1$ IF NO ERROR
4867  013144  000240                          NOP                     ;RETURN HERE IF ERROR
4868  013146  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
4869  013150  000137  013410                  JMP     9$      ;GO TO 9$ IF ERROR WAS FOUND
4870  013154                          1$:
4871  013154  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4872  013162  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4873  013170  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
4874  013176  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4875  013202  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
4876  013204  000240                          NOP                     ;RETURN HERE IF ERROR
4877  013206  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4878  013210  000137  013410                  JMP     9$              ;GO TO 9$ IF ERROR WAS FOUND
4879  013214                          2$:
4880  013214  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
4881  013222  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
4882  013230  012737  000000  001422          MOV     #0,RMMR10       ;RMMR1=0
```

```
4883  013236  004737  044466          JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
4884  013242  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
4885  013244  000240                  NOP                     ;RETURN HERE IF ERROR
4886  013246  104000                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4887  013250  000137  013410          JMP     9$              ;GO TO 9$ IF ERROR WAS FOUND
4888  013254                  3$:
4889  013254  112737  000000  001533  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4890  013262  112737  000200  001534  MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4891  013270  012737  000023  001376  MOV     #PAKACK!GO,RMCS1O       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
4892  013276  004737  044466          JSR     PC,PUT          ;CALL PUT SUBROUTINE
4893  013302  000404                  BR      4$      ;GO TO 4$ IF NO ERROR
4894  013304  000240                  NOP                     ;RETURN HERE TO REPORT AN ERROR
4895  013306  104000                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4896  013310  000137  013410          JMP     9$      ;GO TO 9$ IF ERROR WAS FOUND
4897  013314  004737  044132  4$:     JSR     PC,GETSTS
4898  013320  004737  045026          JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
4899  013324  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4900  013330  000403                  BR      5$      ;GO TO 5$ IF NO ERROR
4901  013332  000240                  NOP                     ;RETURN HERE IF ERROR
4902  013334  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
4903  013336  000424                  BR      9$              ;SKIP REMAINDER OF TEST
4904  013340                  5$:
4905  013340  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4906  013344  000404                  BR      6$                      ;GO TO 6$ IF NO ERROR
4907  013346  000240                  NOP                     ;RETURN HERE IF ERROR
4908  013350  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4909  013352  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4910  013354  000415                  BR      9$              ;SKIP TO NEXT TEST
4911  013356                  6$:
4912  013356  004737  054356          JSR     PC,ACKSTS       ;GO VERIFY PACK ACKNOWLEDGE
4913  013362  000404                  BR      7$                      ;GO TO 7$ IF NO ERROR
4914  013364  000240                  NOP                     ;RETURN HERE IF ERROR
4915  013366  104000                  ERROR                   ;ERROR # DEFINED BY ACKSTS SUBROUTINE
4916  013370  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4917  013372  000406                  BR      9$              ;SKIP TO NEXT TEST
4918
4919  013374                  7$:
4920  013374  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4921  013400  000403                  BR      9$                      ;GO TO 9$ IF NO ERROR
4922  013402  000240                  NOP                     ;RETURN HERE IF ERROR
4923  013404  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4924  013406  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4925  013410                  9$:
4926
4927                          ;;*****************************************************************
4928                          ;*TEST 13        RECALIBRATE TEST
4929
4930                          ;;*****************************************************************
4931  013410                  TST13:
4932  013410  000004                  SCOPE                   ;SCOPE CALL
4933  013412  000240                  NOP                     ;START OF TEST
4934  013414  012737  000001  001206  MOV     #1,$TIMES       ;LOAD ITERATION COUNT
4935  013422  012737  013436  001122  MOV     #1$,$LPADR
4936  013430  012737  013436  001124  MOV     #1$,$LPERR
4937  013436                  1$:
4938  013436  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
4939  013442  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4940  013446  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4941  013452  012737  000013  001226        MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4942  013460  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4943  013464  050020                        .WORD   050020  ;TASK DESCRIPTOR
4944  013466  000404                        BR      5$                      ;GO TO 5$ IF NO ERROR
4945  013470  000240                        NOP                     ;RETURN HERE IF ERROR
4946  013472  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4947  013474  000137  013642                JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4948  013500                        5$:
4949  013500  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
4950  013506  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
4951  013514  012737  000007  001376        MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
4952  013522  004737  044466                JSR     PC,PUT          ;CALL PUT SUBROUTINE
4953  013526  000404                        BR      6$      ;GO TO 6$ IF NO ERROR
4954  013530  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
4955  013532  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
4956  013534  000137  013642                JMP     10$     ;GO TO 10$ IF ERROR WAS FOUND
4957  013540  004737  044132        6$:     JSR     PC,GETSTS       ;GO SETUP FOR STATUS FETCH
4958  013544  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
4959  013550  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4960  013554  000404                        BR      7$      ;GO TO 7$ IF NO ERROR
4961  013556  000240                        NOP                     ;RETURN HERE IF ERROR
4962  013560  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4963  013562  000137  013642                JMP     10$     ;GO TO 10$ IF ERROR WAS FOUND
4964  013566                        7$:
4965  013566  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4966  013572  000405                        BR      8$                      ;GO TO 8$ IF NO ERROR
4967  013574  000240                        NOP                     ;RETURN HERE IF ERROR
4968  013576  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4969  013600  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4970  013602  000137  013642                JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4971  013606                        8$:
4972  013606  004737  055152                JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
4973  013612  000405                        BR      9$                      ;GO TO 9$ IF NO ERROR
4974  013614  000240                        NOP                     ;RETURN HERE IF ERROR
4975  013616  104000                        ERROR                   ;ERROR # DEFINED BY RCLSTS SUBROUTINE
4976  013620  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4977  013622  000137  013642                JMP     10$                     ;GO TO 10$ IF ERROR WAS FOUND
4978  013626                        9$:
4979  013626  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4980  013632  000403                        BR      10$                     ;GO TO 10$ IF NO ERROR
4981  013634  000240                        NOP                     ;RETURN HERE IF ERROR
4982  013636  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4983  013640  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4984  013642                        10$:
4985
4986                                ;;*****************************************************************
4987                                ;*TEST 14        ABORT RECALIBRATE TEST
4988                                ;;*****************************************************************
4989  013642                        TST14:
4990  013642  000004                        SCOPE                   ;SCOPE CALL
4991  013644  000240                        NOP                     ;START OF TEST
4992  013646  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
4993  013654  012737  013670  001122        MOV     #1$,$LPADR
4994  013662  012737  013670  001124        MOV     #1$,$LPERR
```

```
                                        1$:
013670                                          
013670  012706  001100                          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
013674  013700  001276                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
013700  013701  001446                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
013704  012737  000014  001226                  MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                
013712  004737  043216                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
013716  054130                                  .WORD   054130  ;TASK DESCRIPTOR
013720  000404                                  BR      8$                      ;GO TO 8$ IF NO ERROR
013722  000240                                  NOP                     ;RETURN HERE IF ERROR
013724  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
013726  000137  014164                          JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
013732                                  8$:
013732  112737  000014  001533                  MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
013740  112737  000200  001534                  MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
013746  012737  040000  001412                  MOV     #UNS,RMER10             ;RMER1 OUTPUT BUFFER = UNS
013754  004737  044466                          JSR     PC,PUT                  ;WRITE RMER1 VIA PUT SUB
013760  000404                                  BR      9$      ;GO TO 9$ IF NO ERROR
013762  000240                                  NOP                     ;RETURN HERE TO REPORT ERROR
013764  104000                                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
013766  000137  014164                          JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
013772                                  9$:
013772  112737  000000  001533                  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
014000  112737  000200  001534                  MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
014006  012737  000007  001376                  MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
014014  004737  044466                          JSR     PC,PUT          ;CALL PUT SUBROUTINE
014020  000404                                  BR      10$     ;GO TO 10$ IF NO ERROR
014022  000240                                  NOP                     ;RETURN HERE TO REPORT AN ERROR
014024  104000                                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
014026  000137  014164                          JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
014032  004737  044132                  10$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
014036  004737  044216                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
014042  000404                                  BR      11$     ;GO TO 11$ IF NO ERROR
014044  000240                                  NOP                     ;RETURN HERE IF ERROR
014046  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
014050  000137  014164                          JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
014054  032737  020000  001340  11$:    BIT     #PIP,RMDSI      ;DID THE DRIVE RECALIBRATE??
014062  001401                                  BEQ     12$             ;NO!!
014064  104214                                  ERROR   214             ;ERROR SHOULD PREVENT RECALIBRATE
014066  004737  045026                  12$:    JSR     PC,TIMOUT       ;WAIT FOR  COMPLETION
014072  004737  044216                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
014076  000404                                  BR      13$     ;GO TO 13$ IF NO ERROR
014100  000240                                  NOP                     ;RETURN HERE IF ERROR
014102  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
014104  000137  014164                          JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
014110                                  13$:
014110  004737  045212                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
014114  000405                                  BR      14$                     ;GO TO 14$ IF NO ERROR
014116  000240                                  NOP                     ;RETURN HERE IF ERROR
014120  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
014122  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
014124  000137  014164                          JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
014130                                  14$:
014130  004737  061062                          JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
014134  000405                                  BR      141$                    ;GO TO 141$ IF NO ERROR
014136  000240                                  NOP                     ;RETURN HERE IF ERROR
```

```
5051  014140  104000                        ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5052  014142  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5053  014144  000137  014164                JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5054  014150                        141$:
5055  014150  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5056  014154  000403                        BR      15$                     ;GO TO 15$ IF NO ERROR
5057  014156  000240                        NOP                     ;RETURN HERE IF ERROR
5058  014160  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5059  014162  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5060  014164                        15$:
5061
5062                                ;;*****************************************************************
5063                                ;*TEST 15        IVC RECALIBRATE TEST
5064                                ;;*****************************************************************
5065  014164                        TST15:
5066  014164  000004                        SCOPE                   ;SCOPE CALL
5067  014166  000240                        NOP                     ;START OF TEST
5068  014170  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5069  014176  012737  014212  001122        MOV     #1$,$LPADR
5070  014204  012737  014212  001124        MOV     #1$,$LPERR
5071  014212                        1$:
5072  014212  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5073  014216  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5074  014222  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5075  014226  012737  000015  001226        MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5076  014234  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5077  014240  054130                        .WORD   054130  ;TASK DESCRIPTOR
5078  014242  000404                        BR      8$                      ;GO TO 8$ IF NO ERROR
5079  014244  000240                        NOP                     ;RETURN HERE IF ERROR
5080  014246  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5081  014250  000137  014510                JMP     15$                     ;GO TO 15$ IF ERROR WAS FOUND
5082  014254                        8$:
5083  014254  112737  000024  001533        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
5084  014262  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5085  014270  012737  000001  001422        MOV     #DMD,RMMR10     ;RMMR1=DMD
5086  014276  004737  044466                JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
5087  014302  000404                        BR      9$              ;GO TO 9$ IF NO ERROR
5088  014304  000240                        NOP                     ;RETURN HERE IF ERROR
5089  014306  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5090  014310  000137  014510                JMP     15$             ;GO TO 15$ IF ERROR WAS FOUND
5091  014314                        9$:
5092  014314  112737  000024  001533        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
5093  014322  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5094  014330  012737  000000  001422        MOV     #0,RMMR10       ;RMMR1=0
5095  014336  004737  044466                JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
5096  014342  000404                        BR      10$             ;GO TO 10$ IF NO ERROR
5097  014344  000240                        NOP                     ;RETURN HERE IF ERROR
5098  014346  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5099  014350  000137  014510                JMP     15$             ;GO TO 15$ IF ERROR WAS FOUND
5100  014354                        10$:
5101  014354  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5102  014362  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5103  014370  012737  000007  001376        MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
5104  014376  004737  044466                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5105  014402  000404                        BR      11$     ;GO TO 11$ IF NO ERROR
5106  014404  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
```

```
5107  014406  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5108  014410  000137  014510                JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
5109  014414  004737  044132        11$:    JSR     PC,GETST$       ;SETUP FOR STATUS FETCH
5110  014420  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
5111  014424  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5112  014430  000404                        BR      12$     ;GO TO 12$ IF NO ERROR
5113  014432  000240                        NOP                     ;RETURN HERE IF ERROR
5114  014434  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5115  014436  000137  014510                JMP     15$     ;GO TO 15$ IF ERROR WAS FOUND
5116  014442                        12$:
5117  014442  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5118  014446  000405                        BR      13$             ;GO TO 13$ IF NO ERROR
5119  014450  000240                        NOP                     ;RETURN HERE IF ERROR
5120  014452  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5121  014454  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5122  014456  000137  014510                JMP     15$             ;GO TO 15$ IF ERROR WAS FOUND
5123  014462  032737  010000  001370  13$:  BIT     #IVC,RMER2I     ;IVC SHOULD BE SET
5124  014470  001001                        BNE     14$             ;IT IS - GO CHECK SECONDARY ERRORS
5125  014472  104215                        ERROR   215             ;IVC NOT SET DURING RECALIBRATE
5126  014474                        14$:
5127  014474  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5128  014500  000403                        BR      15$             ;GO TO 15$ IF NO ERROR
5129  014502  000240                        NOP                     ;RETURN HERE IF ERROR
5130  014504  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5131  014506  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5132  014510                        15$:
5133
5134                                  ;;*****************************************************************
5135                                  ;*TEST 16        IAE RECALIBRATE TEST
5136                                  ;;*****************************************************************
5137  014510                        TST16:
5138  014510  000004                        SCOPE                   ;SCOPE CALL
5139  014512  000240                        NOP                     ;START OF TEST
5140  014514  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5141  014522  012737  014536  001122        MOV     #1$,$LPADR
5142  014530  012737  014536  001124        MOV     #1$,$LPERR
5143  014536                        1$:
5144  014536  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5145  014542  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5146  014546  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5147  014552  012737  000016  001226        MOV     #16,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
5148  014560  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5149  014564  054130                        .WORD   054130  ;TASK DESCRIPTOR
5150  014566  000404                        BR      70$             ;GO TO 70$ IF NO ERROR
5151  014570  000240                        NOP                     ;RETURN HERE IF ERROR
5152  014572  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5153  014574  000137  014766                JMP     120$            ;GO TO 120$ IF ERROR WAS FOUND
5154  014600                        70$:
5155  014600  012737  177777  001432        MOV     #-1,RMDC0       ;RMDC WILL BE ALL ONES
5156  014606  012737  177777  001404        MOV     #-1,RMDA0       ;RMDA WILL BE ALL ONES
5157  014614  012737  000007  001376        MOV     #RECAL!GO,RMCS10
5158  014622  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO PUT INDEX TABLE
5159  014626  112722  000034                MOVB    #RMDC,(R2)+     ;SETUP PUT INDEX TABLE
5160  014632  112722  000006                MOVB    #RMDA,(R2)+
5161  014636  112722  000000                MOVB    #RMCS1,(R2)+
5162  014642  112722  000200                MOVB    #200,(R2)+
```

```
5163  014646  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5164  014652  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5165  014654  000240                          NOP                     ;RETURN HERE IF ERROR
5166  014656  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5167  014660  000137  014766                  JMP     120$    ;GO TO 120$ IF ERROR WAS FOUND
5168  014664  004737  044132          80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5169  014670  004737  045026                  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
5170  014674  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5171  014700  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5172  014702  000240                          NOP                     ;RETURN HERE IF ERROR
5173  014704  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5174  014706  000137  014766                  JMP     120$    ;GO TO 120$ IF ERROR WAS FOUND
5175  014712                          90$:
5176  014712  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5177  014716  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5178  014720  000240                          NOP                     ;RETURN HERE IF ERROR
5179  014722  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5180  014724  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5181  014726  000137  014766                  JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
5182  014732                          100$:
5183  014732  004737  055152                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
5184  014736  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5185  014740  000240                          NOP                     ;RETURN HERE IF ERROR
5186  014742  104000                          ERROR                   ;ERROR # DEFINED BY RCLSTS SUBROUTINE
5187  014744  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5188  014746  000137  014766                  JMP     120$                    ;GO TO 120$ IF ERROR WAS FOUND
5189  014752                          110$:
5190  014752  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5191  014756  000403                          BR      120$                    ;GO TO 120$ IF NO ERROR
5192  014760  000240                          NOP                     ;RETURN HERE IF ERROR
5193  014762  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5194  014764  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5195  014766                          120$:
5196
5197                                  ;;*****************************************************************
5198                                  ;*TEST 17        RECALIBRATE AT OFFSET
5199                                  ;;*****************************************************************
5200  014766                          TST17:
5201  014766  000004                          SCOPE                   ;SCOPE CALL
5202  014770  000240                          NOP                     ;START OF TEST
5203  014772  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
5204  015000  012737  015014  001122          MOV     #1$,$LPADR
5205  015006  012737  015014  001124          MOV     #1$,$LPERR
5206  015014                          1$:
5207  015014  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5208  015020  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5209  015024  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5210  015030  012737  000017  001226          MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5211
5212  015036  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5213  015042  050020                          .WORD   050020  ;TASK DESCRIPTOR
5214  015044  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
5215  015046  000240                          NOP                     ;RETURN HERE IF ERROR
5216  015050  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5217  015052  000137  015260                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
5218  015056                          5$:
```

```
5219  015056  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5220  015064  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5221  015072  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5222  015100  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5223  015104  000404                          BR      10$     ;GO TO 10$ IF NO ERROR
5224  015106  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5225  015110  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5226  015112  000137  015260                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
5227  015116                          10$:
5228  015116  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5229  015124  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5230  015132  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;RMCS1 OUTPUT BUFFER = RECAL!GO
5231  015140  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5232  015144  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
5233  015146  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5234  015150  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5235  015152  000137  015260                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
5236  015156                          20$:
5237  015156  004737  044132                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5238  015162                          30$:
5239  015162  004737  045026                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5240  015166                          40$:
5241  015166  004737  044216                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5242  015172  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
5243  015174  000240                          NOP                     ;RETURN HERE IF ERROR
5244  015176  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5245  015200  000137  015260                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
5246  015204                          50$:
5247  015204  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5248  015210  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
5249  015212  000240                          NOP                     ;RETURN HERE IF ERROR
5250  015214  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5251  015216  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5252  015220  000137  015260                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
5253  015224                          60$:
5254  015224  004737  055152                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
5255  015230  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
5256  015232  000240                          NOP                     ;RETURN HERE IF ERROR
5257  015234  104000                          ERROR                   ;ERROR # DEFINED BY RCLSTS SUBROUTINE
5258  015236  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5259  015240  000137  015260                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
5260  015244                          70$:
5261  015244  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5262  015250  000403                          BR      80$                     ;GO TO 80$ IF NO ERROR
5263  015252  000240                          NOP                     ;RETURN HERE IF ERROR
5264  015254  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5265  015256  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5266  015260                          80$:
5267                                  ;;*****************************************************************
5268                                  ;#TEST 20       DRIVE CLEAR TEST
5269
5270                                  ;;*****************************************************************
5271  015260                          TST20:
5272  015260  000004                          SCOPE                   ;SCOPE CALL
5273  015262  000240                          NOP                     ;START OF TEST
5274  015264  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
5275  015270  013700  001276              MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5276  015274  013701  001446              MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5277  015300  012737  000020  001226      MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5278  015306  004737  043216              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5279  015312  054130                      .WORD   054130  ;TASK DESCRIPTOR
5280  015314  000404                      BR      2$              ;GO TO 2$ IF NO ERROR
5281  015316  000240                      NOP                     ;RETURN HERE IF ERROR
5282  015320  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5283  015322  000137  015530              JMP     21$             ;GO TO 21$ IF ERROR WAS FOUND
5284  015326                      2$:
5285  015326  012737  000000  001404      MOV     #0,RMDA0
5286  015334  012737  000011  001376      MOV     #DRVCLR!GO,RMCS10
5287  015342  012737  177777  001412      MOV     #-1,RMER10
5288  015350  012737  177777  001440      MOV     #-1,RMER20
5289  015356  042737  004000  001440      BIC     #LSC,RMER20     ;DELETE FOR PASS 2 ETCH
5290  015364  012702  001533              MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
5291  015370  112722  000006              MOVB    #RMDA,(R2)+     ;RMDA CLEARS LBT
5292  015374  112722  000042              MOVB    #RMER2,(R2)+
5293  015400  112722  000014              MOVB    #RMER1,(R2)+
5294  015404  112722  000000              MOVB    #RMCS1,(R2)+
5295  015410  112722  000200              MOVB    #200,(R2)+
5296  015414  004737  044466              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5297  015420  000403                      BR      3$      ;GO TO 3$ IF NO ERROR
5298  015422  000240                      NOP                     ;RETURN HERE IF ERROR
5299  015424  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
5300  015426  000440                      BR      21$             ;ESCAPE IF ERROR
5301  015430  004737  044132      3$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5302  015434  004737  045026              JSR     PC,TIMOUT       ;WAIT FOR DRIVE CLEAR TO COMPLETE
5303  015440  004737  044216              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5304  015444  000403                      BR      4$      ;GO TO 4$ IF NO ERROR
5305  015446  000240                      NOP                     ;RETURN HERE IF ERROR
5306  015450  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
5307  015452  000426                      BR      21$             ;SKIP REMAINDER OF TEST
5308  015454                      4$:
5309  015454  004737  045212              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5310  015460  000405                      BR      5$              ;GO TO 5$ IF NO ERROR
5311  015462  000240                      NOP                     ;RETURN HERE IF ERROR
5312  015464  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5313  015466  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5314  015470  000137  015530              JMP     21$             ;GO TO 21$ IF ERROR WAS FOUND
5315  015474                      5$:
5316  015474  004737  056714              JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR
5317  015500  000405                      BR      6$              ;GO TO 6$ IF NO ERROR
5318  015502  000240                      NOP                     ;RETURN HERE IF ERROR
5319  015504  104000                      ERROR                   ;ERROR # DEFINED BY DRVSTS SUBROUTINE
5320  015506  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5321  015510  000137  015530              JMP     21$             ;GO TO 21$ IF ERROR WAS FOUND
5322  015514                      6$:
5323  015514  004737  046044              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5324  015520  000403                      BR      21$             ;GO TO 21$ IF NO ERROR
5325  015522  000240                      NOP                     ;RETURN HERE IF ERROR
5326  015524  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5327  015526  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5328  015530                      21$:
5329
5330                              ;;*****************************************************************
```

```
                                ;*TEST 21       NOP TEST
                                ;
                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
015530                          TST21:
015530  000004                          SCOPE                   ;SCOPE CALL
015532  000240                          NOP                     ;START OF TEST
015534  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
015540  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
015544  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
015550  012737  000021  001226          MOV     #21,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
015556  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
015562  054130                          .WORD   054130  ;TASK DESCRIPTOR
015564  000404                          BR      70$             ;GO TO 70$ IF NO ERROR
015566  000240                          NOP                     ;RETURN HERE IF ERROR
015570  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015572  000137  015760                  JMP     130$            ;GO TO 130$ IF ERROR WAS FOUND
015576                          70$:
015576  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
015604  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
015612  012737  000001  001376          MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
015620  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
015624  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
015626  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
015630  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
015632  000137  015760                  JMP     130$    ;GO TO 130$ IF ERROR WAS FOUND
015636  004737  044132          80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
015642  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
015646  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
015650  000240                          NOP                     ;RETURN HERE IF ERROR
015652  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
015654  000137  015760                  JMP     130$    ;GO TO 130$ IF ERROR WAS FOUND
015660                          90$:
015660  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
015664  000405                          BR      100$            ;GO TO 100$ IF NO ERROR
015666  000240                          NOP                     ;RETURN HERE IF ERROR
015670  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
015672  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015674  000137  015760                  JMP     130$            ;GO TO 130$ IF ERROR WAS FOUND
015700                          100$:
015700  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
015704  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
015706  000010                          .WORD   DPE             ;MASK FOR RMER2
015710  000405                          BR      110$            ;GO TO 110$ IF NO ERROR
015712  000240                          NOP                     ;RETURN HERE IF ERROR
015714  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
015716  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015720  000137  015760                  JMP     130$            ;GO TO 130$ IF ERROR WAS FOUND
015724                          110$:
015724  004737  061062                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
015730  000405                          BR      120$            ;GO TO 120$ IF NO ERROR
015732  000240                          NOP                     ;RETURN HERE IF ERROR
015734  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
015736  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015740  000137  015760                  JMP     130$            ;GO TO 130$ IF ERROR WAS FOUND
015744                          120$:
015744  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
```

```
                                        BR      130$                    ;GO TO 130$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                130$:
                                ;;*****************************************************************
                                ;*TEST 22        OFFSET TEST
                                ;;*****************************************************************
                                TST22:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #22,$TESTN              ;SET TEST NUMBER IN APT MAIL BOX
                                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      40$                     ;GO TO 40$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
                                40$:
                                        MOVB    #RMCS1,PUTINX           ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
                                        MOV     #OFFSET!GO,RMCS10               ;RMCS1 OUTPUT BUFFER = OFFSET!GO
                                        JSR     PC,PUT                  ;CALL PUT SUBROUTINE
                                        BR      50$     ;GO TO 50$ IF NO ERROR
                                        NOP                             ;RETURN HERE TO REPORT AN ERROR
                                        ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
                                        JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
                                50$:    JSR     PC,GETST$               ;SETUP FOR STATUS FETCH
                                        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     100$    ;GO TO 100$ 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     100$                    ;GO TO 100$ IF ERROR WAS FOUND
                                70$:    BIT     #OM,RMDSI               ;OFFSET MODE ON??
                                        BNE     80$                     ;YES!!
                                        BIC     #^COM,$BDDAT            ;BAD DATA FOR TYPEOUT
                                        MOV     #OM,$GDDAT              ;GOOD DATA FOR TYPEOUT
                                        ERROR   156                     ;OFFSET MODE NOT SET
                                80$:    BIT     #ATA,RMDSI              ;WAS ATTENTION SET ??
                                        BNE     90$                     ;YES!!
                                        MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
                                        BIC     #^CATA,$BDDAT
```

```
5443  016214  012737  100000  001140          MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
5444  016222  104172                          ERROR   172             ;ATTENTION NOT SET
5445  016224                          90$:
5446  016224  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5447  016230  115760                          .WORD   ND!MSK          ;MASK FOR RMER1
5448  016232  000010                          .WORD   DPE             ;MASK FOR RMER2
5449  016234  000405                          BR      91$                     ;GO TO 91$ IF NO ERROR
5450  016236  000240                          NOP                     ;RETURN HERE IF ERROR
5451  016240  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5452  016242  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5453  016244  000137  016304                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
5454  016250                          91$:
5455  016250  004737  061062                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5456  016254  000405                          BR      92$                     ;GO TO 92$ IF NO ERROR
5457  016256  000240                          NOP                     ;RETURN HERE IF ERROR
5458  016260  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5459  016262  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5460  016264  000137  016304                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
5461  016270                          92$:
5462  016270  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5463  016274  000403                          BR      100$                    ;GO TO 100$ IF NO ERROR
5464  016276  000240                          NOP                     ;RETURN HERE IF ERROR
5465  016300  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5466  016302  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5467  016304                          100$:
5468
5469                                  ;;*****************************************************************
5470                                  ;*TEST 23        GO/ATA TEST
5471                                  ;;*****************************************************************
5472  016304                          TST23:
5473  016304  000004                          SCOPE                   ;SCOPE CALL
5474  016306  000240                          NOP                     ;START OF TEST
5475  016310  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5476  016314  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5477  016320  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5478  016324  012737  000023  001226          MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5479
5480  016332  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5481  016336  054130                          .WORD   054130  ;TASK DESCRIPTOR
5482  016340  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5483  016342  000240                          NOP                     ;RETURN HERE IF ERROR
5484  016344  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5485  016346  000137  016724                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5486  016352                          70$:
5487  016352  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5488  016360  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5489  016366  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5490  016374  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
5491  016400  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5492  016402  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
5493  016404  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5494  016406  000137  016724                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5495  016412  004737  044132          80$:    JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
5496  016416  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5497  016422  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5498  016424  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
5499  016426  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5500  016430  000137  016724                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5501  016434  032737  100000  001340  90$:   BIT     #ATA,RMDSI              ;IS ATTENTION SET??
5502  016442  001012                        BNE     100$                    ;YES!!
5503  016444  012737  100000  001140        MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5504  016452  013737  001340  001142        MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5505  016460  042737  077777  001142        BIC     #↑CATA,$BDDAT
5506  016466  104172                        ERROR   172                     ;REPORT ATA NOT SET
5507  016470                          100$:
5508  016470  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5509  016474  000000                        .WORD   0               ;MASK FOR RMER1
5510  016476  000000                        .WORD   0               ;MASK FOR RMER2
5511  016500  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5512  016502  000240                        NOP                     ;RETURN HERE IF ERROR
5513  016504  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5514  016506  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5515  016510  000137  016724                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5516  016514                          110$:
5517  016514  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5518  016522  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5519  016530  012737  000001  001376        MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
5520  016536  004737  044466                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5521  016542  000404                        BR      120$    ;GO TO 120$ IF NO ERROR
5522  016544  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
5523  016546  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5524  016550  000137  016724                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5525  016554                          120$:
5526  016554  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5527  016560  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
5528  016562  000240                        NOP                     ;RETURN HERE IF ERROR
5529  016564  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5530  016566  000137  016724                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5531  016572                          130$:
5532  016572  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5533  016576  000405                        BR      140$                    ;GO TO 140$ IF NO ERROR
5534  016600  000240                        NOP                     ;RETURN HERE IF ERROR
5535  016602  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5536  016604  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5537  016606  000137  016724                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5538  016612  032737  100000  001340  140$:  BIT     #ATA,RMDSI              ;IS ATTENTION RESET??
5539  016620  001411                        BEQ     150$                    ;YES
5540  016622  013737  001340  001142        MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5541  016630  042737  077777  001142        BIC     #↑CATA,$BDDAT
5542  016636  005037  001140                CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
5543  016642  104246                        ERROR   246                     ;REPORT ATA NOT RESET
5544  016644                          150$:
5545  016644  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5546  016650  115760                        .WORD   ND↑MSK          ;MASK FOR RMER1
5547  016652  000010                        .WORD   DPE             ;MASK FOR RMER2
5548  016654  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5549  016656  000240                        NOP                     ;RETURN HERE IF ERROR
5550  016660  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5551  016662  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5552  016664  000137  016724                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5553  016670                          160$:
5554  016670  004737  061062                JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
```

```
5555  016674  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
5556  016676  000240                        NOP                             ;RETURN HERE IF ERROR
5557  016700  104000                        ERROR                           ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5558  016702  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5559  016704  000137  016724                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5560  016710                          170$:
5561  016710  004737  046044                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5562  016714  000403                        BR      180$                    ;GO TO 180$ IF NO ERROR
5563  016716  000240                        NOP                             ;RETURN HERE IF ERROR
5564  016720  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5565  016722  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5566  016724                          180$:
5567
5568                                  ;;*****************************************************************
5569                                  ;*TEST 24       WRITE ATA TEST
5570                                  ;;*****************************************************************
5571  016724                          TST24:
5572  016724  000004                        SCOPE                           ;SCOPE CALL
5573  016726  000240                        NOP                             ;START OF TEST
5574  016730  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
5575  016734  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5576  016740  013701  001446                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5577  016744  012737  000024  001226        MOV     #24,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5578
5579  016752  004737  043216                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5580  016756  054130                        .WORD   054130  ;TASK DESCRIPTOR
5581  016760  000404                        BR      70$                             ;GO TO 70$ IF NO ERROR
5582  016762  000240                        NOP                             ;RETURN HERE IF ERROR
5583  016764  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5584  016766  000137  017270                JMP     150$                            ;GO TO 150$ IF ERROR WAS FOUND
5585  016772                          70$:
5586  016772  112737  000000  001533        MOVB    #RMCS1,PUTINX           ;SETUP PUT INDEX TABLE
5587  017000  112737  000200  001534        MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
5588  017006  012737  000015  001376        MOV     #OFFSET!GO,RMCS10               ;RMCS1 OUTPUT BUFFER = OFFSET!GO
5589  017014  004737  044466                JSR     PC,PUT                  ;CALL PUT SUBROUTINE
5590  017020  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
5591  017022  000240                        NOP                             ;RETURN HERE TO REPORT AN ERROR
5592  017024  104000                        ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
5593  017026  000137  017270                JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5594  017032  004737  044132          80$:  JSR     PC,GETSTS                       ;SETUP FOR STATUS FETCH
5595  017036  004737  044216                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5596  017042  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5597  017044  000240                        NOP                             ;RETURN HERE IF ERROR
5598  017046  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
5599  017050  000137  017270                JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5600  017054  032737  100000  001340  90$:  BIT     #ATA,RMDSI                      ;IS ATTENTION SET??
5601  017062  001012                        BNE     100$                            ;YES!!
5602  017064  013737  001340  001142        MOV     RMDSI,$BDDAT                    ;BAD DATA FOR TYPEOUT
5603  017072  042737  077777  001142        BIC     #↑CATA,$BDDAT
5604  017100  012737  100000  001140        MOV     #ATA,$GDDAT                     ;GOOD DATA FOR TYPEOUT
5605  017106  104172                        ERROR   172                             ;REPORT ATA NOT SET
5606  017110  116102  000001          100$: MOVB    1(R1),R2        ;R2 = ATTENTION BIT
5607  017114  042702  177400                BIC     #↑CATNMSK,R2    ;CLEAR SIGN EXTENSION
5608  017120  000240                        NOP
5609  017122  010237  001414                MOV     R2,RMAS0                        ;PUT RMAS BIT IN OUTPUT BUF
5610  017126  112737  000016  001533        MOVB    #RMAS,PUTINX                    ;WRITE REGISTER INDEX IN TABLE
```

```
5611  017134  112737  000200  001534          MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
5612  017142  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5613  017146  000404                          BR      110$    ;GO TO 110$ IF NO ERROR
5614  017150  000240                          NOP                     ;RETURN HERE IF ERROR
5615  017152  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5616  017154  000137  017270                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5617  017160                          110$:
5618  017160  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5619  017164  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
5620  017166  000240                          NOP                     ;RETURN HERE IF ERROR
5621  017170  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5622  017172  000137  017270                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5623  017176                          120$:
5624  017176  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5625  017202  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
5626  017204  000240                          NOP                     ;RETURN HERE IF ERROR
5627  017206  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5628  017210  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5629  017212  000137  017270                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5630  017216  032737  100000  001340  130$:   BIT     #ATA,RMDSI              ;IS ATTENTION RESET??
5631  017224  001411                          BEQ     140$                    ;YES!!
5632  017226  013737  001340  001142          MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5633  017234  042737  077777  001142          BIC     #^CATA,$BDDAT
5634  017242  005037  001140                  CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
5635  017246  104247                          ERROR   247                     ;REPORT ATA NOT RESET
5636  017250                          140$:
5637  017250  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5638  017254  000000                          .WORD   0               ;MASK FOR RMER1
5639  017256  000000                          .WORD   0               ;MASK FOR RMER2
5640  017260  000403                          BR      150$                    ;GO TO 150$ IF NO ERROR
5641  017262  000240                          NOP                     ;RETURN HERE IF ERROR
5642  017264  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5643  017266  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5644  017270                          150$:
5645
5646                                  ;;*****************************************************************
5647                                  ;*TEST 25        ERROR/ATA TEST
5648                                  ;;*****************************************************************
5649  017270                          TST25:
5650  017270  000004                          SCOPE                   ;SCOPE CALL
5651  017272  000240                          NOP                     ;START OF TEST
5652  017274  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5653  017300  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5654  017304  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5655  017310  012737  000025  001226          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5656
5657  017316  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5658  017322  054130                          .WORD   054130  ;TASK DESCRIPTOR
5659  017324  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
5660  017326  000240                          NOP                     ;RETURN HERE IF ERROR
5661  017330  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5662  017332  000137  017662                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
5663  017336                          70$:
5664  017336  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5665  017344  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5666  017352  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
```

```
5667  017360  004737  044466                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
5668  017364  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5669  017366  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
5670  017370  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5671  017372  000137  017662                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5672  017376  004737  044132            80$:    JSR     PC,GETSTS               ;SETUP FOR STATUS
5673  017402  004737  044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5674  017406  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5675  017410  000240                            NOP                     ;RETURN HERE IF ERROR
5676  017412  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5677  017414  000137  017662                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5678  017420  032737  100000  001340    90$:    BIT     #ATA,RMDSI              ;IS ATA SET??
5679  017426  001012                            BNE     100$                    ;YES!!
5680  017430  013737  001340  001142            MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5681  017436  042737  077777  001142            BIC     #^CATA,$BDDAT
5682  017444  012737  100000  001140            MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5683  017452  104172                            ERROR   172                     ;REPORT ATA NOT SET
5684  017454                            100$:
5685  017454  112737  000014  001533            MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
5686  017462  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
5687  017470  012737  040000  001412            MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
5688  017476  004737  044466                    JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
5689  017502  000404                            BR      110$    ;GO TO 110$ IF NO ERROR
5690  017504  000240                            NOP                     ;RETURN HERE TO REPORT ERROR
5691  017506  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5692  017510  000137  017662                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5693  017514                            110$:
5694  017514  112737  000000  001533            MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5695  017522  112737  000200  001534            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5696  017530  012737  000001  001376            MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
5697  017536  004737  044466                    JSR     PC,PUT          ;CALL PUT SUBROUTINE
5698  017542  000404                            BR      120$    ;GO TO 120$ IF NO ERROR
5699  017544  000240                            NOP                     ;RETURN HERE TO REPORT AN ERROR
5700  017546  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5701  017550  000137  017662                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5702  017554                            120$:
5703  017554  004737  044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5704  017560  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
5705  017562  000240                            NOP                     ;RETURN HERE IF ERROR
5706  017564  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5707  017566  000137  017662                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
5708  017572  032737  100000  001340    130$:   BIT     #ATA,RMDSI              ;IS ATA STILL SET??
5709  017600  001012                            BNE     140$                    ;YES!!
5710  017602  012737  100000  001140            MOV     #ATA,$GDDAT             ;GOOD DATA FOR TYPEOUT
5711  017610  013737  001340  001142            MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
5712  017616  042737  077777  001142            BIC     #^CATA,$BDDAT
5713  017624  104250                            ERROR   250                     ;ATA SHOULD NOT RESET
5714  017626                            140$:
5715  017626  004737  061062                    JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5716  017632  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
5717  017634  000240                            NOP                     ;RETURN HERE IF ERROR
5718  017636  104000                            ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5719  017640  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5720  017642  000137  017662                    JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
5721  017646                            150$:
5722  017646  004737  046044                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
```

```
5723  017652  000403                        BR      160$                    ;GO TO 160$ IF NO ERROR
5724  017654  000240                        NOP                             ;RETURN HERE IF ERROR
5725  017656  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5726  017660  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5727  017662                          160$:
5728
5729                                  ;;*****************************************************************
5730                                  ;*TEST 26       PROGRAM INTERRUPT TEST
5731                                  ;;*****************************************************************
5732  017662                          TST26:
5733  017662  000004                        SCOPE                           ;SCOPE CALL
5734  017664  000240                        NOP                             ;START OF TEST
5735  017666  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
5736  017672  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5737  017676  013701  001446                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5738  017702  012737  000026  001226        MOV     #26,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5739
5740  017710  004737  043216                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5741  017714  054130                        .WORD   054130  ;TASK DESCRIPTOR
5742  017716  000404                        BR      70$                     ;GO TO 70$ IF NO ERROR
5743  017720  000240                        NOP                             ;RETURN HERE IF ERROR
5744  017722  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5745  017724  000137  020330                JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5746
5747  017730  113702  001272          70$:  MOVB    $VECT1,R2               ;R2 = RM03 INTERRUPT ADDRESS
5748  017734  042702  177400                BIC     #^C<377>,R2             ;CLEAR SIGN EXTENSION
5749  017740  011204                        MOV     (R2),R4                 ;SAVE HANDLER ADDRESS
5750  017742  016205  000002                MOV     2(R2),R5                ;SAVE HANDLER PRIORITY
5751  017746  012712  020202                MOV     #150$,(R2)              ;WRITE HANDLER ADDRESS AND
5752  017752  012762  000300  000002        MOV     #PR6,2(R2)              ;PRIORITY FOR THIS TEST
5753  017760  113703  001273                MOVB    $VECT1+1,R3             ;R3 = RM03 INTERRUPT LEVEL
5754  017764  042703  177400                BIC     #^C<377>,R3             ;CLEAR SIGN EXTENSION
5755  017770  000303                        SWAB    R3
5756  017772  005303                        DEC     R3                      ;DECREMENT INTERRUPT LEVEL
5757  017774  100001                        BPL     80$
5758  017776  005003                        CLR     R3
5759  020000  042703  177437          80$:  BIC     #^CPR7,R3
5760  020004  000240                        NOP
5761  020006  012746  000300                MOV     #PR6,-(SP)              ;;PUSH #PR6 ON STACK
5762  020012  012746  020140                MOV     #120$,-(SP)             ;;PUSH #120$ ON STACK
5763  020016  010346                        MOV     R3,-(SP)                ;;PUSH R3 ON STACK
5764  020020  012746  020132                MOV     #110$,-(SP)             ;;PUSH #110$ ON STACK
5765  020024  012737  177777  001414        MOV     #-1,RMAS0               ;WRITE ONES IN RMAS TO
5766  020032  112737  000016  001533        MOVB    #RMAS,PUTINX            ;CLEAR ALL ATTENTIONS
5767  020040  112737  000200  001533        MOVB    #200,PUTINX
5768  020046  004737  044466                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5769  020052  000402                        BR      90$     ;GO TO 90$ IF NO ERROR
5770  020054  000240                        NOP                             ;RETURN HERE IF ERROR
5771  020056  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
5772  020060                          90$:
5773  020060  112737  000000  001533        MOVB    #RMCS1,PUTINX           ;SETUP PUT INDEX TABLE
5774  020066  112737  000200  001534        MOVB    #200,PUTINX+1           ;WRITE TERMINATOR
5775  020074  012737  000115  001376        MOV     #OFFSET!GO!IE,RMCS10    ;RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
5776  020102  004737  044466                JSR     PC,PUT                  ;CALL PUT SUBROUTINE
5777  020106  000402                        BR      100$    ;GO TO 100$ IF NO ERROR
5778  020110  000240                        NOP                             ;RETURN HERE TO REPORT AN ERROR
```

```
5779  020112  104000                        ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
5780  020114  004737  045026        100$:   JSR     PC,TIMOUT                       ;WAIT FOR COMPLETION
5781  020120  004737  044132                JSR     PC,GETSTS                       ;SETUP FOR STATUS
5782  020124  012703  000205                MOV     #205,R3                         ;R3=GROSS TIMEOUT
5783  020130  000002                        RTI                                     ;DROP CP PRIORITY
5784  020132  005303        110$:   DEC     R3                              ;TIMEOUT THE INTERRUPT
5785  020134  100376                        BPL     110$
5786                          ;TIMEOUT BEFORE INTERRUPT
5787  020136  000002                        RTI                                     ;RAISE CP PRIORITY
5788  020140                  120$:
5789  020140  004737  044216                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5790  020144  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
5791  020146  000240                        NOP                             ;RETURN HERE IF ERROR
5792  020150  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
5793  020152  000137  020330                JMP     210$    ;GO TO 210$ IF ERROR WAS FOUND
5794  020156                  130$:
5795  020156  004737  045212                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5796  020162  000405                        BR      140$                            ;GO TO 140$ IF NO ERROR
5797  020164  000240                        NOP                             ;RETURN HERE IF ERROR
5798  020166  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5799  020170  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5800  020172  000137  020330                JMP     210$                            ;GO TO 210$ IF ERROR WAS FOUND
5801  020176  104251        140$:   ERROR   251                             ;REPORT NO INTERRUPT
5802  020200  000423                        BR      180$
5803
5804  020202  022626        150$:   CMP     (SP)+,(SP)+                     ;ADJUST STACK
5805  020204  012716  020212                MOV     #160$,(SP)                      ;CHANGE RETURN ADDRESS
5806  020210  000002                        RTI                                     ;RAISE CP PRIORITY
5807  020212                  160$:
5808  020212  004737  044216                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5809  020216  000404                        BR      170$    ;GO TO 170$ IF NO ERROR
5810  020220  000240                        NOP                             ;RETURN HERE IF ERROR
5811  020222  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
5812  020224  000137  020330                JMP     210$    ;GO TO 210$ IF ERROR WAS FOUND
5813  020230                  170$:
5814  020230  004737  045212                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5815  020234  000405                        BR      180$                            ;GO TO 180$ IF NO ERROR
5816  020236  000240                        NOP                             ;RETURN HERE IF ERROR
5817  020240  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5818  020242  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5819  020244  000137  020330                JMP     210$                            ;GO TO 210$ IF ERROR WAS FOUND
5820  020250                  180$:
5821  020250  004737  061566                JSR     PC,CMPERRSTS            ;CHECK ANY ERRORS NOT MASKED
5822  020254  115760                        .WORD   NDTMSK                  ;MASK FOR RMER1
5823  020256  000010                        .WORD   DPE                     ;MASK FOR RMER2
5824  020260  000405                        BR      190$                            ;GO TO 190$ IF NO ERROR
5825  020262  000240                        NOP                             ;RETURN HERE IF ERROR
5826  020264  104000                        ERROR                           ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5827  020266  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5828  020270  000137  020330                JMP     210$                            ;GO TO 210$ IF ERROR WAS FOUND
5829  020274                  190$:
5830  020274  004737  061062                JSR     PC,STCDRVSTS            ;GO CHECK FOR CHANGES IN DRIVE STATUS
5831  020300  000405                        BR      200$                            ;GO TO 200$ IF NO ERROR
5832  020302  000240                        NOP                             ;RETURN HERE IF ERROR
5833  020304  104000                        ERROR                           ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5834  020306  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
```

```
5835  020310  000137  020330            JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5836  020314                    200$:
5837  020314  004737  046044            JSR     PC,$ECERR       ;GO CHECK FOR SECONDARY ERRORS
5838  020320  000403                    BR      210$                    ;GO TO 210$ IF NO ERROR
5839  020322  000240                    NOP                     ;RETURN HERE IF ERROR
5840  020324  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5841  020326  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5842  020330  010412            210$:   MOV     R4,(R2)
5843  020332  010562  000002            MOV     R5,2(R2)
5844                            ;;*****************************************************************
5845                            ;*TEST 27        INHIBIT INTERRUPT TEST
5846                            ;;*****************************************************************
5847  020336                    TST27:
5848  020336  000004                    SCOPE                   ;SCOPE CALL
5849  020340  000240                    NOP                     ;START OF TEST
5850  020342  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5851  020346  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5852  020352  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5853  020356  012737  000027  001226    MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5854
5855  020364  004737  043216            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5856  020370  054130                    .WORD   054130  ;TASK DESCRIPTOR
5857  020372  000404                    BR      70$                     ;GO TO 70$ IF NO ERROR
5858  020374  000240                    NOP                     ;RETURN HERE IF ERROR
5859  020376  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5860  020400  000137  020670            JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
5861  020404  113702  001272    70$:    MOVB    $VECT1,R2       ;R2 = RM03 INTERRUPT ADDRESS
5862  020410  042702  177400            BIC     #^C<377>,R2     ;CLEAR SIGN EXTENSION
5863  020414  011204                    MOV     (R2),R4                 ;SAVE HANDLER ADDRESS
5864  020416  016205  000002            MOV     2(R2),R5                ;SAVE HANDLER PRIORITY
5865  020422  012712  020634            MOV     #130$,(R2)              ;WRITE HANDLER ADDRESS AND
5866  020426  012762  000300  000002    MOV     #PR6,2(R2)              ;PRIORITY FOR THIS TEST
5867  020434  113703  001273            MOVB    $VECT1+1,R3     ;R3 = RM03 INTERRUPT LEVEL
5868  020440  042703  177400            BIC     #^C<377>,R3     ;CLEAR SIGN EXTENSION
5869  020444  012746  000300            MOV     #PR6,-(SP)      ;;PUSH #PR6 ON STACK
5870  020450  012746  020662            MOV     #180$,-(SP)     ;;PUSH #180$ ON STACK
5871  020454  010346                    MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5872  020456  005316                    DEC     (SP)
5873  020460  100001                    BPL     80$
5874  020462  005016                    CLR     (SP)
5875  020464  042716  177437    80$:    BIC     #^CPR7,(SP)
5876  020470  012746  020622            MOV     #120$,-(SP)     ;;PUSH #120$ ON STACK
5877  020474  010346                    MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5878  020476  012746  020554            MOV     #100$,-(SP)     ;;PUSH #100$ ON STACK
5879  020502  004737  044132            JSR     PC,GETSTS                       ;SETUP FOR STATUS
5880  020506  112737  000000  001533    MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5881  020514  112737  000200  001534    MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5882  020522  012737  000115  001376    MOV     #OFFSET!GO!IE,RMCS10    ;RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
5883  020530  004737  044466            JSR     PC,PUT                  ;CALL PUT SUBROUTINE
5884  020534  000402                    BR      90$     ;GO TO 90$ IF NO ERROR
5885  020536  000240                    NOP                     ;RETURN HERE TO REPORT AN ERROR
5886  020540  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5887  020542  004737  045026    90$:    JSR     PC,TIMOUT               ;WAIT FOR GO=0
5888  020546  012703  000205            MOV     #205,R3                 ;R3=GROSS TIMER
5889  020552  000002                    RTI                             ;MAKE PRIORITY=RM01
5890  020554  005303            100$:   DEC     R3                      ;DELAY
```

```
5891  020556  100376                        BPL     100$
5892  020560  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5893  020566  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5894  020574  012737  000000  001376        MOV     #NOP,RMCS1O     ;RMCS1 OUTPUT BUFFER = NOP
5895  020602  004737  044466                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5896  020606  000402                        BR      110$    ;GO TO 110$ IF NO ERROR
5897  020610  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
5898  020612  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5899  020614  012703  000205        110$:   MOV     #205,R3
5900  020620  000002                        RTI
5901  020622  012712  020636        120$:   MOV     #140$,(R2)              ;MAKE PRIORITY < RM01
5902  020626  005303        125$:   DEC     R3
5903  020630  100376                        BPL     125$
5904  020632  000002                        RTI                     ;NO ERROR
5905  020634  022626        130$:   CMP     (SP)+,(SP)+             ;ADJUST STACK
5906  020636  022626        140$:   CMP     (SP)+,(SP)+             ;ADJUST STACK
5907  020640  012716  020646                MOV     #160$,(SP)
5908  020644  000002                        RTI
5909  020646                160$:
5910  020646  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5911  020652  000402                        BR      170$    ;GO TO 170$ IF NO ERROR
5912  020654  000240                        NOP                     ;RETURN HERE IF ERROR
5913  020656  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5914  020660  104252        170$:   ERROR   252                     ;SHOULD NOT HAVE INTERRUPT
5915  020662  010412        180$:   MOV     R4,(R2)
5916  020664  010562  000002                MOV     R5,2(R2)
5917  020670                210$:
5918                                ;;*****************************************************************
5919                                ;*TEST 30        RETURN TO CENTERLINE TEST
5920                                ;;*****************************************************************
5921  020670                        T30:
5922  020670  000004                        SCOPE                   ;SCOPE CALL
5923  020672  000240                        NOP                     ;START OF TEST
5924  020674  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5925  020700  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5926  020704  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5927  020710  012737  000030  001226        MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5928  020716  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5929  020722  054130                        .WORD   054130  ;TASK DESCRIPTOR
5930  020724  000404                        BR      70$                     ;GO TO 70$ IF NO ERROR
5931  020726  000240                        NOP                     ;RETURN HERE IF ERROR
5932  020730  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5933  020732  000137  021212                JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5934  020736                        70$:
5935  020736  112737  000000  001533        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
5936  020744  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
5937  020752  012737  000017  001376        MOV     #RTC!GO,RMCS1O  ;RMCS1 OUTPUT BUFFER = RTC!GO
5938  020760  004737  044466                JSR     PC,PUT          ;CALL PUT SUBROUTINE
5939  020764  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
5940  020766  000240                        NOP                     ;RETURN HERE TO REPORT AN ERROR
5941  020770  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
5942  020772  000137  021212                JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5943  020776  004737  044132        80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
5944  021002  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
5945  021006  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5946  021012  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
```

```
5947  021014  000240                          NOP                     ;RETURN HERE IF ERROR
5948  021016  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5949  021020  000137  021212                  JMP     150$    ;GO TO 150$ IF ERROR WAS FOUND
5950  021024                          90$:
5951  021024  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5952  021030  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5953  021032  000240                          NOP                     ;RETURN HERE IF ERROR
5954  021034  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5955  021036  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5956  021040  000137  021212                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5957  021044  032737  000001  001340  100$:   BIT     #OM.RMDSI       ;OFFSET MODE OFF??
5958  021052  001411                          BEQ     110$            ;YES!!
5959  021054  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
5960  021060  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
5961  021066  042737  177776  001142          BIC     #^COM,$BDDAT
5962  021074  104157                          ERROR   157             ;OFFSET MODE NOT RESET
5963  021076  032737  100000  001340  110$:   BIT     #ATA,RMDSI      ;WAS ATTENTION SET ??
5964  021104  001012                          BNE     120$            ;YES!!
5965  021106  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
5966  021114  042737  077777  001142          BIC     #^CATA,$BDDAT
5967  021122  012737  100000  001140          MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
5968  021130  104171                          ERROR   171             ;"ATA" NOT SET DURING RTC
5969  021132                          120$:
5970  021132  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
5971  021136  115760                          .WORD   ND^MSK          ;MASK FOR RMER1
5972  021140  000010                          .WORD   DPE             ;MASK FOR RMER2
5973  021142  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
5974  021144  000240                          NOP                     ;RETURN HERE IF ERROR
5975  021146  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
5976  021150  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5977  021152  000137  021212                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5978  021156                          130$:
5979  021156  004737  061062                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
5980  021162  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
5981  021164  000240                          NOP                     ;RETURN HERE IF ERROR
5982  021166  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
5983  021170  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5984  021172  000137  021212                  JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
5985  021176                          140$:
5986  021176  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5987  021202  000403                          BR      150$                    ;GO TO 150$ IF NO ERROR
5988  021204  000240                          NOP                     ;RETURN HERE IF ERROR
5989  021206  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5990  021210  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5991  021212                          150$:
5992
5993                                  ;;*****************************************************************
5994                                  ;*TEST 31        READ IN PRESET TEST
5995
5996                                  ;;*****************************************************************
5997  021212                          TST31:
5998  021212  000004                          SCOPE                   ;SCOPE CALL
5999  021214  000240                          NOP                     ;START OF TEST
6000  021216  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6001  021222  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6002  021226  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
 6003  021232  012737  000031  001226          MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 6004
 6005  021240  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 6006  021244  054130                          .WORD   054130  ;TASK DESCRIPTOR
 6007  021246  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
 6008  021250  000240                          NOP                     ;RETURN HERE IF ERROR
 6009  021252  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 6010  021254  000137  021574                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 6011  021260                          70$:
 6012  021260  012702  001533                  MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
 6013  021264  112722  000006                  MOVB    #RMDA,(R2)+
 6014  021270  112722  000034                  MOVB    #RMDC,(R2)+
 6015  021274  112722  000032                  MOVB    #RMOF,(R2)+
 6016  021300  112722  000000                  MOVB    #RMCS1,(R2)+
 6017  021304  112722  000200                  MOVB    #200,(R2)+
 6018  021310  012737  177777  001404          MOV     #-1,RMDA0
 6019  021316  012737  177777  001432          MOV     #-1,RMDC0
 6020  021324  012737  016200  001430          MOV     #FMT16!ECI!HCI!OFD,RMOF0
 6021  021332  012737  000021  001376          MOV     #RIP!GO,RMCS10
 6022  021340  004737  044466                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 6023  021344  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
 6024  021346  000240                          NOP                     ;RETURN HERE IF ERROR
 6025  021350  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 6026  021352  000137  021574                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 6027  021356  004737  044132          80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
 6028  021362  004737  045026                  JSR     PC,TIMOUT       ;WAIT FOR RIP TO COMPLETE
 6029  021366  004737  044216                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 6030  021372  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
 6031  021374  000240                          NOP                     ;RETURN HERE IF ERROR
 6032  021376  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 6033  021400  000137  021574                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 6034  021404                          90$:
 6035  021404  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 6036  021410  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
 6037  021412  000240                          NOP                     ;RETURN HERE IF ERROR
 6038  021414  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 6039  021416  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6040  021420  000137  021574                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 6041  021424  013737  001360  001142  100$:   MOV     RMOFI,$BDDAT    ;IS RMOF RESET??
 6042  021432  042737  161577  001142          BIC     #^C<FMT16!ECI!HCI!OFD>,$BDDAT
 6043  021440  001403                          BEQ     110$            ;YES!!
 6044  021442  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
 6045  021446  104160                          ERROR   160             ;RMOF NOT CLEARED BY RIP
 6046  021450  005737  001334          110$:   TST     RMDAI           ;IS RMDA RESET??
 6047  021454  001406                          BEQ     120$            ;YES!!
 6048  021456  013737  001334  001142          MOV     RMDAI,$BDDAT    ;BAD DATA FOR TYPEOUT
 6049  021464  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
 6050  021470  104161                          ERROR   161             ;RMDA NOT CLEARED BY RIP
 6051  021472  005737  001362          120$:   TST     RMDCI           ;IS RMDC RESET??
 6052  021476  001406                          BEQ     130$            ;YES!!
 6053  021500  013737  001362  001142          MOV     RMDCI,$BDDAT    ;BAD DATA FOR TYPEOUT
 6054  021506  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
 6055  021512  104162                          ERROR   162             ;RMDC NOT CLEARED BY RIP
 6056  021514                          130$:
 6057  021514  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 6058  021520  115760                          .WORD   ND1MSK          ;MASK FOR RMER1
```

```
                                        .WORD   DPE             ;MASK FOR RMER2
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
                                140$:
                                        JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
                                        BR      150$                    ;GO TO 150$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
                                150$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      160$                    ;GO TO 160$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                160$:

                                ;;*******************************************************************
                                ;*TEST 32        RMDC CLEAR OFFSET TEST
                                ;;*******************************************************************
                                TST32:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      70$                     ;GO TO 70$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
                                70$:
                                        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
                                        MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
                                        JSR     PC,PUT          ;CALL PUT SUBROUTINE
                                        BR      80$     ;GO TO 80$ IF NO ERROR
                                        NOP                     ;RETURN HERE TO REPORT AN ERROR
                                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                                        JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
                                80$:    JSR     PC,GETSTS       ;SETUP FOR STATUS
                                        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
                                        BR      90$     ;GO TO 90$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR DEFINED BY GET SUB
                                        JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
                                90$:
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      100$                    ;GO TO 100$ IF NO ERROR
```

```
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     140$                            ;GO TO 140$ IF ERROR WAS FOUND
                                100$:   BIT     #OM,RMDSI               ;OFFSET ON??
                                        BNE     110$                    ;YES
                                        MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
                                        BIC     #^COM,$BDDAT
                                        MOV     #OM,$GDDAT              ;GOOD DATA FOR TYPEOUT
                                        ERROR   156                     ;OFFSET MODE NOT SET
                                        BR      140$                    ;SKIP REST OF TEST
                                110$:
                                        MOV     #0,RMDC0                ;CYLINDER = 0
                                        MOVB    #RMDC,PUTINX                    ;SETUP REGISTER INDEX TABLE
                                        MOVB    #200,PUTINX+1                   ;TERMINATE TABLE
                                        JSR     PC,PUT                          ;WRITE RMDC USING PUT SUB
                                        BR      120$                    ;GO TO 120$ IF NO ERROR
                                        NOP                                     ;RETURN HERE IF ERROR
                                        ERROR                                   ;ERROR NUMBER DEFINED BY PUT
                                        JMP     140$                            ;GO TO 140$ IF ERROR
                                120$:
                                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
                                        BR      130$    ;GO TO 130$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR DEFINED BY GET SUB
                                        JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
                                130$:   BIT     #OM,RMDSI               ;DID OFFSET MODE RESET??
                                        BEQ     140$
                                        MOV     RMDSI,$BDDAT            ;BAD DATA FOR TYPEOUT
                                        BIC     #^COM,$BDDAT
                                        CLR     $GDDAT                  ;GOOD DATA FOR TYPEOUT
                                        ERROR   253                     ;WRITING RMDC DIDNT CLEAR OM
                                140$:

                                ;;*****************************************************************
                                ;*TEST 33       ILLEGAL FUNCTION TEST
                                ;*****************************************************************
                                TST33:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #33,$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
```

```
6171  022176  000240                          NOP                     ;RETURN HERE IF ERROR
6172  022200  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
6173  022202  000137  022676                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6174  022206                          20$:
6175
6176                                  ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
6177                                  ;CYLINDER
6178  022206  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6179  022214  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6180  022222  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
6181  022230  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6182  022234  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
6183  022236  000240                          NOP                     ;RETURN HERE IF ERROR
6184  022240  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6185  022242  000137  022676                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6186  022246                          30$:
6187  022246  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6188  022254  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6189  022262  012737  001401  001422          MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6190  022270  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6191  022274  000404                          BR      35$             ;GO TO 35$ IF NO ERROR
6192  022276  000240                          NOP                     ;RETURN HERE IF ERROR
6193  022300  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6194  022302  000137  022676                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6195  022306                          35$:
6196
6197                                  ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
6198  022306  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6199  022314  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6200  022322  012737  000023  001376          MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6201  022330  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
6202  022334  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
6203  022336  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
6204  022340  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6205  022342  000137  022676                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6206  022346                          40$:
6207  022346  012737  000001  001376          MOV     #GO,RMCS10
6208  022354  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6209  022360  012737  106126  001402          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6210  022366  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6211  022374  005037  001404                  CLR     RMDA0           ;CLEAR DISK ADDRESS
6212  022400  005037  001432                  CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
6213  022404  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BHT FORMAT
6214  022412  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6215  022416  112723  000004                  MOVB    #RMBA,(R3)+
6216  022422  112723  000002                  MOVB    #RMWC,(R3)+
6217  022426  112723  000032                  MOVB    #RMOF,(R3)+
6218  022432  112723  000006                  MOVB    #RMDA,(R3)+
6219  022436  112723  000034                  MOVB    #RMDC,(R3)+
6220  022442  112723  000000                  MOVB    #RMCS1,(R3)+
6221  022446  112713  000200                  MOVB    #200,(R3)
6222  022452  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6223  022456  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
6224  022460  000240                          NOP                     ;RETURN HERE IF ERROR
6225  022462  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6226  022464  000137  022676                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
```

```
6227  022470  004737  045026          50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
6228  022474  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6229  022500  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
6230  022502  000240                          NOP                     ;RETURN HERE IF ERROR
6231  022504  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6232  022506  000137  022676                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6233  022512                          60$:
6234  022512  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6235  022516  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
6236  022520  000240                          NOP                     ;RETURN HERE IF ERROR
6237  022522  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6238  022524  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6239  022526  000137  022676                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6240  022532  016237  067254  001140  70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
6241  022540  042737  177776  001140          BIC     #^CILF,$GDDAT
6242  022546  013737  001342  001142          MOV     RMERII,$BDDAT   ;BAD DATA FOR TYPEOUT
6243  022554  042737  177776  001142          BIC     #^CILF,$BDDAT
6244  022562  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ILF STATUS CORRECT?
6245  022570  001402                          BEQ     80$             ;YES!!
6246  022572  104254                          ERROR   254             ;REPORT INCORRECT ILF STATUS
6247  022574  000440                          BR      110$
6248  022576  016237  067254  001140  80$:    MOV     FNCDTB(R2),$GDDAT
6249  022604  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
6250  022612  001403                          BEQ     90$             ;NO!!
6251  022614  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
6252  022622  042737  077777  001140  90$:    BIC     #^CATA,$GDDAT
6253  022630  013737  001340  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
6254  022636  042737  077777  001142          BIC     #^CATA,$BDDAT
6255  022644  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6256  022652  001402                          BEQ     100$            ;YES!!
6257  022654  104255                          ERROR   255             ;INCORRECT ATA STATUS
6258  022656  000407                          BR      110$
6259  022660  062702  000002          100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
6260  022664  022702  000076                  CMP     #ILF76,R2       ;DONE??
6261  022670  103402                          BLO     110$            ;YES !!
6262  022672  000137  022170                  JMP     10$             ;TEST NEXT FUNCTION
6263  022676                          110$:
6264                                  ;;*****************************************************************
6265                                  ;*TEST 34       INVALID COMMAND TEST
6266                                  ;;*****************************************************************
6267  022676                          TST34:
6268  022676  000004                          SCOPE                   ;SCOPE CALL
6269  022700  000240                          NOP                     ;START OF TEST
6270  022702  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6271  022706  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6272  022712  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6273  022716  012737  000034  001226          MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6274
6275  022724  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6276  022730  040000                          .WORD   040000  ;TASK DESCRIPTOR
6277  022732  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
6278  022734  000240                          NOP                     ;RETURN HERE IF ERROR
6279  022736  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6280  022740  000137  023422                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6281  022744  004737  044132          5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
6282  022750  012702  000000                  MOV     #NOP,R2
```

```
6283   022754                          10$:
6284   022754  004737  053360                  JSR     PC,CNTCLR
6285   022760  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
6286   022762  000240                          NOP             ;RETURN HERE IF ERROR
6287   022764  104000                          ERROR           ;ERROR NUMBER DEFINED BY SUB
6288   022766  000137  023422                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6289   022772                          20$:
6290
6291                                   ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
6292                                   ;CYLINDER
6293   022772  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6294   023000  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6295   023006  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
6296   023014  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6297   023020  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
6298   023022  000240                          NOP                     ;RETURN HERE IF ERROR
6299   023024  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6300   023026  000137  023422                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6301   023032                          30$:
6302   023032  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6303   023040  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6304   023046  012737  001401  001422          MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6305   023054  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6306   023060  000404                          BR      35$             ;GO TO 35$ IF NO ERROR
6307   023062  000240                          NOP                     ;RETURN HERE IF ERROR
6308   023064  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6309   023066  000137  023422                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6310   023072                          35$:
6311
6312
6313                                   ;VOLUME VALID IS LEFT RESET FOR THE TEST
6314
6315   023072                          40$:
6316   023072  012737  000001  001376          MOV     #GO,RMCS10
6317   023100  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6318   023104  012737  106126  001402          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6319   023112  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6320   023120  005037  001404                  CLR     RMDA0           ;CLEAR DISK ADDRESS
6321   023124  005037  001432                  CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
6322   023130  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BHT FORMAT
6323   023136  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6324   023142  112723  000004                  MOVB    #RMBA,(R3)+
6325   023146  112723  000002                  MOVB    #RMWC,(R3)+
6326   023152  112723  000032                  MOVB    #RMOF,(R3)+
6327   023156  112723  000006                  MOVB    #RMDA,(R3)+
6328   023162  112723  000034                  MOVB    #RMDC,(R3)+
6329   023166  112723  000000                  MOVB    #RMCS1,(R3)+
6330   023172  112713  000200                  MOVB    #200,(R3)
6331   023176  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6332   023202  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
6333   023204  000240                          NOP             ;RETURN HERE IF ERROR
6334   023206  104000                          ERROR           ;ERROR DEFINED BY PUT SUB
6335   023210  000137  023422                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6336   023214  004737  045026          50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
6337   023220  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6338   023224  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
```

```
023226  000240                          NOP                     ;RETURN HERE IF ERROR
023230  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
023232  000137  023422                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
023236                          60$:
023236  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
023242  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
023244  000240                          NOP                     ;RETURN HERE IF ERROR
023246  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
023250  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
023252  000137  023422                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
023256  016237  067254  001140  70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
023264  042737  167777  001140          BIC     #^CIVC,$GDDAT
023272  013737  001370  001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
023300  042737  167777  001142          BIC     #^CIVC,$BDDAT
023306  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS IVC STATUS CORRECT?
023314  001402                          BEQ     80$             ;YES!!
023316  104216                          ERROR   216             ;REPORT INCORRECT IVC STATUS
023320  000440                          BR      110$
023322  016237  067254  001140  80$:    MOV     FNCDTB(R2),$GDDAT
023330  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
023336  001403                          BEQ     90$             ;NO!!
023340  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
023346  042737  077777  001140  90$:    BIC     #^CATA,$GDDAT
023354  013737  001340  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
023362  042737  077777  001142          BIC     #^CATA,$BDDAT
023370  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
023376  001402                          BEQ     100$            ;YES!!
023400  104255                          ERROR   255             ;INCORRECT ATA STATUS
023402  000407                          BR      110$
023404  062702  000002          100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
023410  022702  000076                  CMP     #ILF76,R2       ;DONE??
023414  103402                          BLO     110$            ;YES !!
023416  000137  022754                  JMP     10$             ;TEST NEXT FUNCTION
023422                          110$:
                        ;;*****************************************************************
                        ;*TEST 35       INVALID ADDRESS ERROR TEST
                        ;;*****************************************************************
023422                  TST35:
023422  000004                          SCOPE                   ;SCOPE CALL
023424  000240                          NOP                     ;START OF TEST
023426  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
023432  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
023436  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
023442  012737  000035  001226          MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
023450  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
023454  040000                          .WORD   040000  ;TASK DESCRIPTOR
023456  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
023460  000240                          NOP                     ;RETURN HERE IF ERROR
023462  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
023464  000137  024212                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
023470  004737  044132          5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
023474  012702  000000                  MOV     #NOP,R2
023500                          10$:
023500  004737  053360                  JSR     PC,CNTCLR
023504  000404                          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$:

                                ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
                                        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
                                        MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
                                        JSR     PC,PUT          ;CALL PUT SUBROUTINE
                                        BR      40$     ;GO TO 40$ IF NO ERROR
                                        NOP                     ;RETURN HERE TO REPORT AN ERROR
                                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                                        JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
                                40$:
                                        MOV     #GO,RMCS10
                                        BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
                                        MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
                                        MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
                                        MOV     #-1,RMDA0       ;USE INVALID DISK ADDRESS
                                        MOV     #-1,RMDC0       ;USE INVALID CYLINDER ADDRESS
                                        MOV     #FMT16,RMOF0    ;16 BIT FORMAT
                                        MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
                                        MOVB    #RMBA,(R3)+
                                        MOVB    #RMWC,(R3)+
                                        MOVB    #RMOF,(R3)+
                                        MOVB    #RMDA,(R3)+
                                        MOVB    #RMDC,(R3)+
                                        MOVB    #RMCS1,(R3)+
                                        MOVB    #200,(R3)
                                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
                                        BR      50$     ;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
```

```
6451  024004  004737  045026          50$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
6452  024010  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6453  024014  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
6454  024016  000240                          NOP                     ;RETURN HERE IF ERROR
6455  024020  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6456  024022  000137  024212                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6457  024026                          60$:
6458  024026  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6459  024032  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
6460  024034  000240                          NOP                     ;RETURN HERE IF ERROR
6461  024036  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6462  024040  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6463  024042  000137  024212                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6464  024046  016237  067254  001140  70$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
6465  024054  042737  175777  001140          BIC     #↑CIAE,$GDDAT
6466  024062  013737  001342  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
6467  024070  042737  175777  001142          BIC     #↑CIAE,$BDDAT
6468  024076  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS IAE STATUS CORRECT?
6469  024104  001402                          BEQ     80$             ;YES!!
6470  024106  104217                          ERROR   217             ;REPORT INCORRECT IAE STATUS
6471  024110  000440                          BR      110$
6472  024112  016237  067254  001140  80$:    MOV     FNCDTB(R2),$GDDAT
6473  024120  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
6474  024126  001403                          BEQ     90$             ;NO!!
6475  024130  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
6476  024136  042737  077777  001140  90$:    BIC     #↑CATA,$GDDAT
6477  024144  013737  001340  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
6478  024152  042737  077777  001142          BIC     #↑CATA,$BDDAT
6479  024160  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6480  024166  001402                          BEQ     100$            ;YES!!
6481  024170  104255                          ERROR   255             ;INCORRECT ATA STATUS
6482  024172  000407                          BR      110$
6483  024174  062702  000002          100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
6484  024200  022702  000076                  CMP     #ILF76,R2       ;DONE??
6485  024204  103402                          BLO     110$            ;YES !!
6486  024206  000137  023500                  JMP     10$             ;TEST NEXT FUNCTION
6487  024212                          110$:
6488                                  ;;*****************************************************************
6489                                  ;*TEST 36       WRITE LOCK ERROR TEST
6490                                  ;;*****************************************************************
6491  024212                          ↑ST36:
6492  024212  000004                          SCOPE                   ;SCOPE CALL
6493  024214  000240                          NOP                     ;START OF TEST
6494  024216  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6495  024222  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6496  024226  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6497  024232  012737  000036  001226          MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6498
6499  024240  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6500  024244  040000                          .WORD   040000  ;TASK DESCRIPTOR
6501  024246  000404                          BR      5$                      ;GO TO 5$ IF NO ERROR
6502  024250  000240                          NOP                     ;RETURN HERE IF ERROR
6503  024252  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6504  024254  000137  024776                  JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6505  024260  004737  044132          5$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
6506  024264  012702  000000                  MOV     #NOP,R2
```

```
6507  024270                          10$:
6508  024270  004737  053360                  JSR     PC,CNTCLR
6509  024274  000404                          BR      20$     ;GO TO 20$ IF NO ERROR
6510  024276  000240                          NOP             ;RETURN HERE IF ERROR
6511  024300  104000                          ERROR           ;ERROR NUMBER DEFINED BY SUB
6512  024302  000137  024776                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6513  024306                          20$:
6514
6515                                  ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
6516                                  ;CYLINDER, AND DIAGNOSTIC WRITE LOCK
6517  024306  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6518  024314  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6519  024322  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
6520  024330  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6521  024334  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
6522  024336  000240                          NOP                     ;RETURN HERE IF ERROR
6523  024340  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6524  024342  000137  024776                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6525  024346                          30$:
6526  024346  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6527  024354  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6528  024362  012737  001411  001422          MOV     #DMD!MUR!MOC!MWP,RMMR10 ;RMMR1=DMD!MUR!MOC!MWP
6529  024370  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6530  024374  000404                          BR      35$             ;GO TO 35$ IF NO ERROR
6531  024376  000240                          NOP                     ;RETURN HERE IF ERROR
6532  024400  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6533  024402  000137  024776                  JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6534  024406                          35$:
6535
6536                                  ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
6537  024406  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6538  024414  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6539  024422  012737  000023  001376          MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6540  024430  004737  044466                  JSR     PC,PUT          ;CALL PUT SUBROUTINE
6541  024434  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
6542  024436  000240                          NOP                     ;RETURN HERE TO REPORT AN ERROR
6543  024440  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6544  024442  000137  024776                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6545  024446                          40$:
6546  024446  012737  000001  001376          MOV     #GO,RMCS10
6547  024454  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6548  024460  012737  106126  001402          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6549  024466  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6550  024474  005037  001404                  CLR     RMDA0           ;CLEAR DISK ADDRESS
6551  024500  005037  001432                  CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
6552  024504  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BHT FORMAT
6553  024512  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6554  024516  112723  000004                  MOVB    #RMBA,(R3)+
6555  024522  112723  000002                  MOVB    #RMWC,(R3)+
6556  024526  112723  000032                  MOVB    #RMOF,(R3)+
6557  024532  112723  000006                  MOVB    #RMDA,(R3)+
6558  024536  112723  000034                  MOVB    #RMDC,(R3)+
6559  024542  112723  000000                  MOVB    #RMCS1,(R3)+
6560  024546  112713  000200                  MOVB    #200,(R3)
6561  024552  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6562  024556  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
```

```
6563  024560  000240                        NOP                             ;RETURN HERE IF ERROR
6564  024562  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
6565  024564  000137  024776                JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6566  024570  004737  045026        50$:    JSR     PC,TIMOUT               ;WAIT FOR GO TO RESET
6567  024574  004737  044216                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6568  024600  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
6569  024602  000240                        NOP                             ;RETURN HERE IF ERROR
6570  024604  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
6571  024606  000137  024776                JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6572  024612                        60$:
6573  024612  004737  045212                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6574  024616  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
6575  024620  000240                        NOP                             ;RETURN HERE IF ERROR
6576  024622  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6577  024624  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6578  024626  000137  024776                JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6579  024632  016237  067254  001140  70$:  MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
6580  024640  042737  173777  001140        BIC     #^CWLE,$GDDAT
6581  024646  013737  001342  001142        MOV     RMER11,$BDDAT   ;BAD DATA FOR TYPEOUT
6582  024654  042737  173777  001142        BIC     #^CWLE,$BDDAT
6583  024662  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS WLE STATUS CORRECT?
6584  024670  001402                        BEQ     80$             ;YES!!
6585  024672  104220                        ERROR   220             ;REPORT INCORRECT WLE STATUS
6586  024674  000440                        BR      110$
6587  024676  016237  067254  001140  80$:  MOV     FNCDTB(R2),$GDDAT
6588  024704  032737  040000  001340        BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
6589  024712  001403                        BEQ     90$             ;NO!!
6590  024714  052737  100000  001140        BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
6591  024722  042737  077777  001140  90$:  BIC     #^CATA,$GDDAT
6592  024730  013737  001340  001142        MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
6593  024736  042737  077777  001142        BIC     #^CATA,$BDDAT
6594  024744  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6595  024752  001402                        BEQ     100$            ;YES!!
6596  024754  104255                        ERROR   255             ;INCORRECT ATA STATUS
6597  024756  000407                        BR      110$
6598  024760  062702  000002        100$:   ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
6599  024764  022702  000076                CMP     #ILF76,R2       ;DONE??
6600  024770  103402                        BLO     110$            ;YES !!
6601  024772  000137  024270                JMP     10$             ;TEST NEXT FUNCTION
6602  024776                        110$:
6603                                ;;*****************************************************************
6604                                ;*TEST 37        OPI TEST
6605                                ;;*****************************************************************
6606  024776                        TST37:
6607  024776  000004                        SCOPE                   ;SCOPE CALL
6608  025000  000240                        NOP                     ;START OF TEST
6609  025002  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6610  025006  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6611  025012  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6612  025016  012737  000037  001226        MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6613
6614  025024  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6615  025030  040000                        .WORD   040000  ;TASK DESCRIPTOR
6616  025032  000404                        BR      5$                      ;GO TO 5$ IF NO ERROR
6617  025034  000240                        NOP                             ;RETURN HERE IF ERROR
6618  025036  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
6619  025040  000137  025622            JMP     110$                    ;GO TO 110$ IF ERROR WAS FOUND
6620  025044  004737  044132      5$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
6621  025050  012702  000000            MOV     #NOP,R2
6622  025054                      10$:
6623  025054  004737  053360            JSR     PC,CNTCLR
6624  025060  000404                    BR      20$     ;GO TO 20$ IF NO ERROR
6625  025062  000240                    NOP                     ;RETURN HERE IF ERROR
6626  025064  104000                    ERROR                   ;ERROR NUMBER DEFINED BY SUB
6627  025066  000137  025622            JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6628  025072                      20$:
6629
6630                              ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
6631                              ;CYLINDER
6632  025072  112737  000024  001533      MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6633  025100  112737  000200  001534      MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6634  025106  012737  000001  001422      MOV     #DMD,RMMR10     ;RMMR1=DMD
6635  025114  004737  044466            JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6636  025120  000404                    BR      30$             ;GO TO 30$ IF NO ERROR
6637  025122  000240                    NOP                     ;RETURN HERE IF ERROR
6638  025124  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6639  025126  000137  025622            JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6640  025132                      30$:
6641  025132  112737  000024  001533      MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6642  025140  112737  000200  001534      MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6643  025146  012737  001401  001422      MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6644  025154  004737  044466            JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6645  025160  000404                    BR      35$             ;GO TO 35$ IF NO ERROR
6646  025162  000240                    NOP                     ;RETURN HERE IF ERROR
6647  025164  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6648  025166  000137  025622            JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6649  025172                      35$:
6650
6651                              ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
6652  025172  112737  000000  001533      MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6653  025200  112737  000200  001534      MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6654  025206  012737  000023  001376      MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6655  025214  004737  044466            JSR     PC,PUT          ;CALL PUT SUBROUTINE
6656  025220  000404                    BR      36$     ;GO TO 36$ IF NO ERROR
6657  025222  000240                    NOP                     ;RETURN HERE TO REPORT AN ERROR
6658  025224  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6659  025226  000137  025622            JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6660  025232                      36$:
6661  025232  112737  000024  001533      MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6662  025240  112737  000200  001534      MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6663  025246  012737  000401  001422      MOV     #DMD!MOC,RMMR10 ;RMMR1=DMD!MOC
6664  025254  004737  044466            JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6665  025260  000404                    BR      40$             ;GO TO 40$ IF NO ERROR
6666  025262  000240                    NOP                     ;RETURN HERE IF ERROR
6667  025264  104000                    ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6668  025266  000137  025622            JMP     110$            ;GO TO 110$ IF ERROR WAS FOUND
6669  025272                      40$:
6670  025272  012737  000001  001376      MOV     #GO,RMCS10
6671  025300  050237  001376            BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6672  025304  012737  106126  001402      MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6673  025312  012737  177777  001400      MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6674  025320  005037  001404            CLR     RMDA0           ;CLEAR DISK ADDRESS
```

```
6675  025324  005037  001432                  CLR     RMDCO                   ;CLEAR CYLINDER ADDRESS
6676  025330  012737  010000  001430          MOV     #FMT16,RMOFO            ;16 BHT FORMAT
6677  025336  012703  001533                  MOV     #PUTINX,R3              ;WRITE REGISTER INDEX TABLE
6678  025342  112723  000004                  MOVB    #RMBA,(R3)+
6679  025346  112723  000002                  MOVB    #RMWC,(R3)+
6680  025352  112723  000032                  MOVB    #RMOF,(R3)+
6681  025356  112723  000006                  MOVB    #RMDA,(R3)+
6682  025362  112723  000034                  MOVB    #RMDC,(R3)+
6683  025366  112723  000000                  MOVB    #RMCS1,(R3)+
6684  025372  112713  000200                  MOVB    #200,(R3)
6685  025376  004737  044466                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6686  025402  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
6687  025404  000240                          NOP                             ;RETURN HERE IF ERROR
6688  025406  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
6689  025410  000137  025622                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6690  025414  004737  045026          50$:    JSR     PC,TIMOUT               ;WAIT FOR GO TO RESET
6691  025420  004737  044216                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6692  025424  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
6693  025426  000240                          NOP                             ;RETURN HERE IF ERROR
6694  025430  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
6695  025432  000137  025622                  JMP     110$    ;GO TO 110$ IF ERROR WAS FOUND
6696  025436                          60$:
6697  025436  004737  045212                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6698  025442  000405                          BR      70$                             ;GO TO 70$ IF NO ERROR
6699  025444  000240                          NOP                             ;RETURN HERE IF ERROR
6700  025446  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6701  025450  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6702  025452  000137  025622                  JMP     110$                            ;GO TO 110$ IF ERROR WAS FOUND
6703  025456  016237  067254  001140  70$:    MOV     FNCDTB(R2),$GDDAT               ;GET ENTRY FROM FUNCTION CODE TABLE
6704  025464  042737  157777  001140          BIC     #↑COPI,$GDDAT
6705  025472  013737  001342  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
6706  025500  042737  157777  001142          BIC     #↑COPI,$BDDAT
6707  025506  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS OPI STATUS CORRECT?
6708  025514  001402                          BEQ     80$                     ;YES!!
6709  025516  104221                          ERROR   221                     ;REPORT INCORRECT OPI STATUS
6710  025520  000440                          BR      110$
6711  025522  016237  067254  001140  80$:    MOV     FNCDTB(R2),$GDDAT
6712  025530  032737  040000  001340          BIT     #ERR,RMDSI              ;WAS AN ERROR DETECTED??
6713  025536  001403                          BEQ     90$                     ;NO!!
6714  025540  052737  100000  001140          BIS     #ATA,$GDDAT             ;YES - ATA SHOULD BE ON
6715  025546  042737  077777  001140  90$:    BIC     #↑CATA,$GDDAT
6716  025554  013737  001340  001142          MOV     RMDSI,$BDDAT            ;GET DRIVE'S ATTENTION
6717  025562  042737  077777  001142          BIC     #↑CATA,$BDDAT
6718  025570  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
6719  025576  001402                          BEQ     100$                    ;YES!!
6720  025600  104255                          ERROR   255                     ;INCORRECT ATA STATUS
6721  025602  000407                          BR      110$
6722  025604  062702  000002          100$:   ADD     #2,R2                   ;GO TO NEXT FUNCTION CODE
6723  025610  022702  000076                  CMP     #ILF76,R2               ;DONE??
6724  025614  103402                          BLO     110$                    ;YES !!
6725  025616  000137  025054                  JMP     10$                     ;TEST NEXT FUNCTION
6726  025622                          110$:
6727                                  ;;*****************************************************************
6728                                  ;*TEST 40        ERROR ABORT TESTS
6729                                  ;;*****************************************************************
6730  025622                          TST40:
```

```
6731  025622  000004                          SCOPE                   ;SCOPE CALL
6732  025624  000240                          NOP                     ;START OF TEST
6733  025626  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6734  025632  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6735  025636  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6736  025642  012737  000040  001226          MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6737
6738  025650  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6739  025654  054130                          .WORD   054130  ;TASK DESCRIPTOR
6740  025656  000404                          BR      10$                     ;GO TO 10$ IF NO ERROR
6741  025660  000240                          NOP                     ;RETURN HERE IF ERROR
6742  025662  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6743  025664  000137  026160                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6744  025670                          10$:
6745  025670  004737  044132                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6746  025674  012702  000000                  MOV     #NOP,R2         ;R2 = FUNCTION CODE
6747  025700                          20$:
6748  025700  004737  053360                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
6749  025704  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6750  025706  000240                          NOP                     ;RETURN HERE IF ERROR
6751  025710  104000                          ERROR                   ;ERROR # DEFINED BY CNTCLR SUBROUTINE
6752  025712  000137  026160                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6753  025716                          30$:
6754  025716  112737  000014  001533          MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
6755  025724  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6756  025732  012737  040000  001412          MOV     #%UNS,RMER10    ;RMER1 OUTPUT BUFFER = #UNS
6757  025740  004737  044466                  JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
6758  025744  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
6759  025746  000240                          NOP                     ;RETURN HERE TO REPORT ERROR
6760  025750  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6761  025752  000137  026160                  JMP     80$     ;GO TO 80$ IF ERROR WAS FOUND
6762  025756                          40$:
6763  025756  012737  000001  001376          MOV     #GO,RMCS10
6764  025764  050237  001376                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
6765  025770  012737  106126  001402          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
6766  025776  012737  177777  001400          MOV     #<^C1+1>,RMWC0  ;DUMMY WORD COUNT
6767  026004  012737  000000  001404          MOV     #0,RMDA0        ;CLEAR DISK ADDRESS
6768  026012  012737  000000  001432          MOV     #0,RMDC0        ;CLEAR CYLINDER ADDRESS
6769  026020  012737  010000  001430          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
6770  026026  012703  001533                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
6771  026032  112723  000004                  MOVB    #RMBA,(R3)+
6772  026036  112723  000002                  MOVB    #RMWC,(R3)+
6773  026042  112723  000032                  MOVB    #RMOF,(R3)+
6774  026046  112723  000006                  MOVB    #RMDA,(R3)+
6775  026052  112723  000034                  MOVB    #RMDC,(R3)+
6776  026056  112723  000000                  MOVB    #RMCS1,(R3)+
6777  026062  112713  000200                  MOVB    #200,(R3)
6778  026066  004737  044466                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6779  026072  000404                          BR      45$                     ;GO TO 45$ IF NO ERROR
6780  026074  000240                          NOP                     ;RETURN HERE IF ERROR
6781  026076  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6782  026100  000137  026160                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6783  026104                          45$:
6784  026104  004737  045026                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6785  026110  004737  044216                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6786  026114  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
```

```
6787  026116  000240                          NOP                     ;RETURN HERE IF ERROR
6788  026120  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6789  026122  000137  026160                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6790  026126                          50$:
6791  026126  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6792  026132  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
6793  026134  000240                          NOP                     ;RETURN HERE IF ERROR
6794  026136  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6795  026140  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6796  026142  000137  026160                  JMP     80$                     ;GO TO 80$ IF ERROR WAS FOUND
6797  026146                          60$:
6798  026146                          70$:
6799  026146  062702  000002                  ADD     #2,R2           ;ADVANCE FUNCTION CODE
6800  026152  022702  000076                  CMP     #ILF76,R2       ;DONE ALL COMMANDS
6801  026156  103250                          BHIS    20$             ;NO !!
6802  026160                          80$:
6803                                  ;;*****************************************************************
6804                                  ;*TEST 41        RMR TEST
6805                                  ;;*****************************************************************
6806  026160                          TST41:
6807  026160  000004                          SCOPE                   ;SCOPE CALL
6808  026162  000240                          NOP                     ;START OF TEST
6809  026164  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6810  026170  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6811  026174  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6812  026200  012737  000041  001226          MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6813
6814  026206  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6815  026212  040000                          .WORD   040000  ;TASK DESCRIPTOR
6816  026214  000404                          BR      10$                     ;GO TO 10$ IF NO ERROR
6817  026216  000240                          NOP                     ;RETURN HERE IF ERROR
6818  026220  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6819  026222  000137  026620                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6820  026226                          10$:
6821  026226  004737  044132                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6822  026232                          20$:
6823  026232  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6824  026240  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6825  026246  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
6826  026254  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6827  026260  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
6828  026262  000240                          NOP                     ;RETURN HERE IF ERROR
6829  026264  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6830  026266  000137  026620                  JMP     100$            ;GO TO 100$ IF ERROR WAS FOUND
6831  026272                          30$:
6832  026272  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6833  026300  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6834  026306  012737  001401  001422          MOV     #DMD!MUR!MOC,RMMR10     ;RMMR1=DMD!MUR!MOC
6835  026314  004737  044466                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
6836  026320  000404                          BR      40$             ;GO TO 40$ IF NO ERROR
6837  026322  000240                          NOP                     ;RETURN HERE IF ERROR
6838  026324  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
6839  026326  000137  026620                  JMP     100$            ;GO TO 100$ IF ERROR WAS FOUND
6840  026332                          40$:
6841  026332  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6842  026340  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
```

```
6843  026346  012737  000023  001376          MOV     #PAKACK!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = PAKACK!GO
6844  026354  004737  044466                  JSR     PC,PUT                  ;CALL PUT SUBROUTINE
6845  026360  000404                          BR      50$     ;GO TO 50$ IF NO ERROR
6846  026362  000240                          NOP                             ;RETURN HERE TO REPORT AN ERROR
6847  026364  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
6848  026366  000137  026620                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6849  026372                          50$:
6850  026372  004737  044216                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6851  026376  000404                          BR      60$                     ;GO TO 60$ IF NO ERROR
6852  026400  000240                          NOP                     ;RETURN HERE IF ERROR
6853  026402  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6854  026404  000137  026620                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6855  026410                          60$:
6856  026410  000240                          NOP
6857  026412  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
6858  026420  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
6859  026426  012737  041401  001422          MOV     #DMD!MUR!MOC!DBEN,RMMR10        ;RMMR1=DMD!MUR!MOC!DBEN
6860  026434  004737  044466                  JSR     PC,PUT                  ;GO WRITE RMMR1 VIA SUB
6861  026440  000404                          BR      70$                     ;GO TO 70$ IF NO ERROR
6862  026442  000240                          NOP                             ;RETURN HERE IF ERROR
6863  026444  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
6864  026446  000137  026620                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6865  026452                          70$:
6866  026452  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6867  026460  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6868  026466  012737  000001  001376          MOV     #NOP!GO,RMCS10  ;RMCS1 OUTPUT BUFFER = NOP!GO
6869  026474  004737  044466                  JSR     PC,PUT                  ;CALL PUT SUBROUTINE
6870  026500  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6871  026502  000240                          NOP                             ;RETURN HERE TO REPORT AN ERROR
6872  026504  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
6873  026506  000137  026620                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6874  026512                          80$:
6875  026512  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
6876  026520  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
6877  026526  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;RMCS1 OUTPUT BUFFER = OFFSET!GO
6878  026534  004737  044466                  JSR     PC,PUT                  ;CALL PUT SUBROUTINE
6879  026540  000404                          BR      85$     ;GO TO 85$ IF NO ERROR
6880  026542  000240                          NOP                             ;RETURN HERE TO REPORT AN ERROR
6881  026544  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
6882  026546  000137  026620                  JMP     100$    ;GO TO 100$ IF ERROR WAS FOUND
6883  026552                          85$:
6884  026552  004737  044216                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6885  026556  000404                          BR      90$                     ;GO TO 90$ IF NO ERROR
6886  026560  000240                          NOP                     ;RETURN HERE IF ERROR
6887  026562  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6888  026564  000137  026620                  JMP     100$                    ;GO TO 100$ IF ERROR WAS FOUND
6889  026570                          90$:
6890  026570  000240                          NOP
6891  026572  032737  000004  001342          BIT     #RMR,RMER1I             ;IS RMR SET ??
6892  026600  001007                          BNE     100$                    ;YES !!
6893  026602  012737  000004  001140          MOV     #RMR,$GDDAT             ;EXPECTED STATUS
6894  026610  013737  001342  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6895  026616  104222                          ERROR   222                     ;RMR DID NOT SET
6896  026620                          100$:
6897                                  ;;;*****************************************************************
6898                                  ;*TEST 42        PARITY ERROR TEST
```

```
                                ;;*****************************************************************
                                TST42:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   040000  ;TASK DESCRIPTOR
                                        BR      10$                     ;GO TO 10$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
                                10$:
                                        MOV     #1,R2                   ;R2 = DATA PATTERN
                                20$:
                                        JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                                        BR      30$                     ;GO TO 30$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY CNTCLR SUBROUTINE
                                        JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
                                30$:
                                        MOVB    (R1),R3                 ;SETUP RMCS2
                                        BIC     #^CUNTMSK,R3
                                        BIS     #PAT,R3
                                        MOV     R3,RMCS20               ;OUTPUT VALUE TO RMCS2
                                        MOV     R2,RMDA0                ;VALUE TO RMDA
                                        MOV     #PUTINX,R3              ;WRITE REGISTER OUTPUT INDEX
                                        MOVB    #RMCS2,(R3)+
                                        MOVB    #RMDA,(R3)+
                                        MOVB    #200,(R3)+

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      40$                     ;GO TO 40$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
                                40$:
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      50$                     ;GO TO 50$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     70$                     ;GO TO 70$ IF ERROR WAS FOUND
                                50$:
                                        BIT     #PAR,RMER1I             ;IS PARITY ERROR SET ??
                                        BNE     60$                     ;YES !!
                                        MOV     #PAR,$GDDAT             ;EXPECTED STATUS
                                        MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
                                        MOV     R2,$TMP0                ;DATA PATTERN
                                        ERROR   223                     ;PAR DID NOT SET
                                60$:
                                        ASL     R2                      ;ADVANCE DATA PATTERN
                                        BNE     20$                     ;BRANCH IF NOT DONE
                                70$:
```

```
6955                                    ;;*****************************************************************
6956                                    ;*TEST 43        ILLEGAL REGISTER TEST
6957                                    ;;*****************************************************************
6958  027044                            TST43:
6959  027044  000004                           SCOPE                   ;SCOPE CALL
6960  027046  000240                           NOP                     ;START OF TEST
6961  027050  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6962  027054  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6963  027060  013701  001446                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6964  027064  012737  000043  001226           MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6965
6966  027072  004737  043216                   JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6967  027076  040000                           .WORD   040000  ;TASK DESCRIPTOR
6968  027100  000404                           BR      10$             ;GO TO 10$ IF NO ERROR
6969  027102  000240                           NOP                     ;RETURN HERE IF ERROR
6970  027104  104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6971  027106  000137  027372                   JMP     120$            ;GO TO 120$ IF ERROR WAS FOUND
6972  027112  005005                    10$:    CLR     R5              ;R5 = EXPECTED STATUS
6973  027114  004737  044132                   JSR     PC,GETSTS       ;SETUP FOR STATUS
6974  027120  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
6975  027124  013746  000006                   MOV     ERRVEC+2,-(SP)  ;;PUSH ERRVEC+2 ON STACK
6976  027130  012737  027370  000004           MOV     #110$,ERRVEC    ;SETUP FOR BUS TIMEOUT
6977  027136  012737  000300  000006           MOV     #PR6,ERRVEC+2
6978  027144  005002                           CLR     R2              ;R2=REGISTER INDEX
6979  027146                            20$:
6980  027146  004737  053360                   JSR     PC,CNTCLR
6981  027152  000404                           BR      30$     ;GO TO 30$ IF NO ERROR
6982  027154  000240                           NOP                     ;RETURN HERE IF ERROR
6983  027156  104000                           ERROR                   ;ERROR NUMBER DEFINED BY SUB
6984  027160  000137  027304                   JMP     70$     ;GO TO 70$ IF ERROR WAS FOUND
6985  027164  010003                    30$:    MOV     R0,R3           ;R3=REGISTER ADDRESS
6986  027166  060203                           ADD     R2,R3
6987  027170  005013                           CLR     (R3)            ;CLEAR THE REGISTER
6988  027172  004737  051706                   JSR     PC,DEVSEL       ;GO SELECT DEVICE
6989  027176  000404                           BR      35$             ;GO TO 35$ IF NO ERROR
6990  027200  000240                           NOP                     ;RETURN HERE IF ERROR
6991  027202  104000                           ERROR                   ;ERROR # DEFINED BY DEVSEL SUBROUTINE
6992  027204  000137  027372                   JMP     120$            ;GO TO 120$ IF ERROR WAS FOUND
6993  027210                            35$:
6994  027210  004737  044216                   JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6995  027214  000404                           BR      40$     ;GO TO 40$ IF NO ERROR
6996  027216  000240                           NOP                     ;RETURN HERE IF ERROR
6997  027220  104000                           ERROR                   ;ERROR DEFINED BY GET SUB
6998  027222  000137  027304                   JMP     70$     ;GO TO 70$ IF ERROR WAS FOUND
6999  027226                            40$:
7000  027226  004737  045212                   JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7001  027232  000405                           BR      50$             ;GO TO 50$ IF NO ERROR
7002  027234  000240                           NOP                     ;RETURN HERE IF ERROR
7003  027236  104000                           ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7004  027240  004736                           JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7005  027242  000137  027304                   JMP     70$             ;GO TO 70$ IF ERROR WAS FOUND
7006  027246  010537  001140            50$:    MOV     R5,$GDDAT
7007  027252  013737  001342  001142           MOV     RMER1I,$BDDAT   ;GET DRIVE'S ILR STATUS
7008  027260  042737  177775  001142           BIC     #^C1LR,$BDDAT
7009  027266  023737  001140  001142           CMP     $GDDAT,$BDDAT   ;IS ILR STATUS OK??
7010  027274  001403                           BEQ     70$             ;YES!!
```

```
                                        MOV     R2,$TMP0        ;SAVE R2 FOR ERROR DATA
                                        ERROR   256
                                ;70$:   ADD     #2,R2
                                ;       CMP     #RMBAE,R2       ;IS NEXT REGISTER RMBAE??
                                ;       BNE     80$             ;NO!!
                                ;       MOV     #A16!A17,RMCS1(R0)      ;SET A16 & A17
                                ;       MOV     RMBAE(R0),R3    ;IS THIS AN RH70??
                                ;       BIC     #^C<BIT0!BIT1>,R3       ;CLEAR ALL BUT ADDRESS BITS
                                ;       CMP     #BIT0!BIT1,R3   ;ARE ADDRESS BITS ON ??
                                ;       BEQ     100$            ;YES
                                ;       MOV     #ILR,R5         ;NO - EXPECT ILR=1
                                ;       BR      100$
                                ;80$:   CMP     #RMCS3+2,R2     ;SHOULD ILR BE SET??
                                ;       BNE     90$             ;NO!!
                                ;       MOV     #ILR,R5         ;YES!!
                                ;       BR      100$
                                ;90$:   CMP     #76,R2          ;DONE??
                                ;       BLO     120$            ;YES!!
                                ;100$:  BR      20$
                                ;THE FOLLOWING CODING FOR HANDLE RH70
                                ;RH 70 REGISTER NUMBERS IS EITHER 22 OR 32
000002                          70$:    ADD     #2,R2           ;INCREMENT THE REGISTER ADDRESS
000050                                  CMP     #50,R2          ;TIME TO CHECK THE EXTEND ADDRESS REG ?
                                        BHI     20$             ;BRANCH IF NOT
                                        BLO     80$             ;BRANCH IF ALREADY SET UP
000002                                  MOV     #ILR,R5         ;EXCEPT ILR HAPPEND
000000                                  MOV     #A16!A17,RMCS1(R0)      ;SET EXTEMD ADDRESS BIT
000054                                  MOV     #54,R2          ;SET ADDRESS TO 54,IF 22 REGISTERS
000050                                  MOV     50(R0),R3       ;IS THIS 22 REG RH70 ?
177774                                  BIC     #177774,R3      ;LEFT ONLY BIT 0 AND BIT 1
000003                                  CMP     #BIT0!BIT1,R3   ;BIT 0 AND BIT 1 SET AT THE SAME TIME ?
                                        BEQ     80$             ;BRANCH IF SO
000050                                  MOV     #50,R2          ;OTHERWISE SET ADDRESS TO 50
000074                          80$:    CMP     #74,R2          ;ALL REGISTERS CHECKED ?
                                        BLOS    120$            ;BRANCH IF SO
                                        BR      20$             ;NEXT REGISTER

                                110$:   CMP     (SP)+,(SP)+
                                120$:
000006                                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
000004                                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                ;;*****************************************************************
                                ;#TEST 44       SEEK LAST CYLINDER

                                ;;*****************************************************************
                                ^ST44:
000004                                  SCOPE                   ;SCOPE CALL
000240                                  NOP                     ;START OF TEST
000001  001206                         MOV     #1,$TIMES       ;LOAD ITERATION COUNT
027430  001122                         MOV     #1$,$LPADR
027430  001124                         MOV     #1$,$LPERR
                                1$:
001100                                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
001276                                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
001446                                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
000044  001226                         MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
7067  027452                        10$:
7068  027452  004737  043216              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7069  027456  054130                      .WORD   054130  ;TASK DESCRIPTOR
7070  027460  000404                      BR      80$             ;GO TO 80$ IF NO ERROR
7071  027462  000240                      NOP                     ;RETURN HERE IF ERROR
7072  027464  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7073  027466  000137  027704              JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7074  027472                        80$:
7075  027472  012737  000000  001404      MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7076  027500  012737  001466  001432      MOV     #822.,RMDC0     ;CYLINDER = 822
7077  027506  012737  000005  001376      MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
7078  027514  012702  001533              MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7079  027520  112722  000034              MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7080  027524  112722  000006              MOVB    #RMDA,(R2)+
7081  027530  112722  000000              MOVB    #RMCS1,(R2)+
7082  027534  112722  000200              MOVB    #200,(R2)+      ;WRITE TERMINATOR
7083  027540  004737  044466              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7084  027544  000404                      BR      90$     ;GO TO 90$ IF NO ERROR
7085  027546  000240                      NOP                     ;RETURN HERE IF ERROR
7086  027550  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
7087  027552  000137  027704              JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7088  027556  004737  044132        90$:  JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7089  027562  004737  045026              JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
7090  027566  004737  044216              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7091  027572  000404                      BR      100$    ;GO TO 100$ IF NO ERROR
7092  027574  000240                      NOP                     ;RETURN HERE IF ERROR
7093  027576  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
7094  027600  000137  027704              JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7095  027604                        100$:
7096  027604  004737  045212              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7097  027610  000405                      BR      110$            ;GO TO 110$ IF NO ERROR
7098  027612  000240                      NOP                     ;RETURN HERE IF ERROR
7099  027614  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7100  027616  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7101  027620  000137  027704              JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7102  027624                        110$:
7103  027624  004737  052120              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7104  027630  000405                      BR      120$            ;GO TO 120$ IF NO ERROR
7105  027632  000240                      NOP                     ;RETURN HERE IF ERROR
7106  027634  104000                      ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7107  027636  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7108  027640  000137  027704              JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7109  027644                        120$:
7110  027644  004737  061566              JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7111  027650  115760                      .WORD   ND!MSK          ;MASK FOR RMER1
7112  027652  000010                      .WORD   DPE             ;MASK FOR RMER2
7113  027654  000405                      BR      130$            ;GO TO 130$ IF NO ERROR
7114  027656  000240                      NOP                     ;RETURN HERE IF ERROR
7115  027660  104000                      ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7116  027662  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7117  027664  000137  027704              JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7118  027670                        130$:
7119  027670  004737  046044              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7120  027674  000403                      BR      140$            ;GO TO 140$ IF NO ERROR
7121  027676  000240                      NOP                     ;RETURN HERE IF ERROR
7122  027700  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
```

```
7123  027702  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7124  027704                          140$:
7125                                  ;;*****************************************************************
7126                                  ;#TEST 45       SEEK FIRST CYLINDER
7127
7128                                  ;;*****************************************************************
7129  027704                          TST45:
7130  027704  000004                        SCOPE                   ;SCOPE CALL
7131  027706  000240                        NOP                     ;START OF TEST
7132  027710  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7133  027716  012737  027732  001122        MOV     #1$,$LPADR
7134  027724  012737  027732  001124        MOV     #1$,$LPERR
7135  027732                          1$:
7136  027732  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7137  027736  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7138  027742  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7139  027746  012737  000045  001226        MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7140  027754                          10$:
7141  027754  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7142  027760  054130                        .WORD   054130  ;TASK DESCRIPTOR
7143  027762  000404                        BR      80$             ;GO TO 80$ IF NO ERROR
7144  027764  000240                        NOP                     ;RETURN HERE IF ERROR
7145  027766  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7146  027770  000137  030206                JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7147  027774                          80$:
7148  027774  012737  000000  001404        MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7149  030002  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER = 0
7150  030010  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
7151  030016  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7152  030022  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7153  030026  112722  000006                MOVB    #RMDA,(R2)+
7154  030032  112722  000000                MOVB    #RMCS1,(R2)+
7155  030036  112722  000200                MOVB    #200,(R2)+      ;WRITE TERMINATOR
7156  030042  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7157  030046  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7158  030050  000240                        NOP                     ;RETURN HERE IF ERROR
7159  030052  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7160  030054  000137  030206                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7161  030060  004737  044132          90$:  JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7162  030064  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
7163  030070  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7164  030074  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
7165  030076  000240                        NOP                     ;RETURN HERE IF ERROR
7166  030100  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7167  030102  000137  030206                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7168  030106                          100$:
7169  030106  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7170  030112  000405                        BR      110$            ;GO TO 110$ IF NO ERROR
7171  030114  000240                        NOP                     ;RETURN HERE IF ERROR
7172  030116  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7173  030120  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7174  030122  000137  030206                JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
7175  030126                          110$:
7176  030126  004737  052120                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7177  030132  000405                        BR      120$            ;GO TO 120$ IF NO ERROR
7178  030134  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
7179  030136  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7180  030140  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7181  030142  000137  030206                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7182  030146                        120$:
7183  030146  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7184  030152  115760                        .WORD   ND!MSK          ;MASK FOR RMER1
7185  030154  000010                        .WORD   DPE             ;MASK FOR RMER2
7186  030156  000405                        BR      130$                    ;GO TO 130$ IF NO ERROR
7187  030160  000240                        NOP                     ;RETURN HERE IF ERROR
7188  030162  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7189  030164  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7190  030166  000137  030206                JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7191  030172                        130$:
7192  030172  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7193  030176  000403                        BR      140$                    ;GO TO 140$ IF NO ERROR
7194  030200  000240                        NOP                     ;RETURN HERE IF ERROR
7195  030202  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7196  030204  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7197  030206                        140$:
7198                                ;;*****************************************************************
7199                                ;*TEST 46        SEEK PRIME CYLINDERS
7200
7201                                ;;*****************************************************************
7202  030206                        TST46:
7203  030206  000004                        SCOPE                   ;SCOPE CALL
7204  030210  000240                        NOP                     ;START OF TEST
7205  030212  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7206  030220  012737  030234  001122        MOV     #1$,$LPADR
7207  030226  012737  030234  001124        MOV     #1$,$LPERR
7208  030234                        1$:
7209  030234  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7210  030240  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7211  030244  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7212  030250  012737  000046  001226        MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7213  030256  012737  000001  001432        MOV     #1,RMDC0        ;FIRST CYLINDER WILL BE 1
7214  030264                        10$:
7215  030264  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7216  030270  054130                        .WORD   054130  ;TASK DESCRIPTOR
7217  030272  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
7218  030274  000240                        NOP                     ;RETURN HERE IF ERROR
7219  030276  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7220  030300  000137  030534                JMP     150$                    ;GO TO 150$ IF ERROR WAS FOUND
7221  030304  012737  000000  001404  80$:   MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7222  030312  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
7223  030320  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7224  030324  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7225  030330  112722  000006                MOVB    #RMDA,(R2)+
7226  030334  112722  000000                MOVB    #RMCS1,(R2)+
7227  030340  112722  000200                MOVB    #200,(R2)+      ;WRITE TERMINATOR
7228  030344  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7229  030350  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7230  030352  000240                        NOP                     ;RETURN HERE IF ERROR
7231  030354  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7232  030356  000137  030510                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7233  030362  004737  044132        90$:    JSR     PC,GETST$       ;SETUP FOR STATUS FETCH
7234  030366  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
```

```
7235  030372  004737  044216            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7236  030376  000404                    BR      100$    ;GO TO 100$ IF NO ERROR
7237  030400  000240                    NOP                     ;RETURN HERE IF ERROR
7238  030402  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
7239  030404  000137  030510            JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
7240  030410                    100$:
7241  030410  004737  045212            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7242  030414  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
7243  030416  000240                    NOP                     ;RETURN HERE IF ERROR
7244  030420  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7245  030422  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7246  030424  000137  030510            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7247  030430                    110$:
7248  030430  004737  052120            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7249  030434  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
7250  030436  000240                    NOP                     ;RETURN HERE IF ERROR
7251  030440  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7252  030442  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7253  030444  000137  030510            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7254  030450                    120$:
7255  030450  004737  061566            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7256  030454  115760                    .WORD   NDTMSK          ;MASK FOR RMER1
7257  030456  000010                    .WORD   DPE             ;MASK FOR RMER2
7258  030460  000405                    BR      130$                    ;GO TO 130$ IF NO ERROR
7259  030462  000240                    NOP                     ;RETURN HERE IF ERROR
7260  030464  104000                    ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7261  030466  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7262  030470  000137  030510            JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
7263  030474                    130$:
7264  030474  004737  046044            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7265  030500  000403                    BR      140$                    ;GO TO 140$ IF NO ERROR
7266  030502  000240                    NOP                     ;RETURN HERE IF ERROR
7267  030504  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7268  030506  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7269  030510  013746  001432    140$:   MOV     RMDCO,-(SP)     ;SHIFT TO NEXT PRIME CYLINDER
7270  030514  006316                    ASL     (SP)
7271  030516  011637  001432            MOV     (SP),RMDCO
7272  030522  022726  001000            CMP     #512.,(SP)+     ;IS THE TEST DONE??
7273  030526  103402                    BLO     150$            ;YES!!
7274  030530  000137  030264            JMP     10$             ;GO DO NEXT CYLINDER
7275  030534                    150$:
7276
7277
7278                              ;;*****************************************************************
7279                              ;*TEST 47        SEEK ZERO DIFFERENCE
7280
7281                              ;;*****************************************************************
7282  030534                    ST47:
7283  030534  000004                    SCOPE                   ;SCOPE CALL
7284  030536  000240                    NOP                     ;START OF TEST
7285  030540  012737  000001  001206    MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7286  030546  012737  030562  001122    MOV     #1$,$LPADR
7287  030554  012737  030562  001124    MOV     #1$,$LPERR
7288  030562                    1$:
7289  030562  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7290  030566  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
```

```
7291  030572  013701  001446              MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7292  030576  012737  000047  001226      MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7293  030604                          10$:
7294  030604  004737  043216              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7295  030610  054130                      .WORD   054130  ;TASK DESCRIPTOR
7296  030612  000404                      BR      80$                     ;GO TO 80$ IF NO ERROR
7297  030614  000240                      NOP                     ;RETURN HERE IF ERROR
7298  030616  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7299  030620  000137  031056              JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7300  030624                          80$:
7301  030624  012703  000001              MOV     #1,R3           ;R3 = NUMBER OF SEEKS
7302  030630  012737  000000  001432      MOV     #0,RMDC0        ;CYLINDER = 0
7303  030636  012737  000000  001404      MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7304  030644  012737  000005  001376      MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
7305  030652  012702  001533              MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
7306  030656  112722  000006              MOVB    #RMDA,(R2)+     ;WRITE REGISTER INDEX TABLE
7307  030662  112722  000034              MOVB    #RMDC,(R2)+
7308  030666  112722  000000              MOVB    #RMCS1,(R2)+
7309  030672  112722  000200              MOVB    #200,(R2)+      ;TERMINATE TABLE
7310  030676                          90$:
7311  030676  004737  044466              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7312  030702  000404                      BR      100$    ;GO TO 100$ IF NO ERROR
7313  030704  000240                      NOP                     ;RETURN HERE IF ERROR
7314  030706  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
7315  030710  000137  031056              JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7316  030714  004737  044132          100$:JSR    PC,GETSTS       ;SETUP FOR READING STATUS
7317  030720  004737  045026              JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
7318  030724  004737  044216              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7319  030730  000404                      BR      120$    ;GO TO 120$ IF NO ERROR
7320  030732  000240                      NOP                     ;RETURN HERE IF ERROR
7321  030734  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
7322  030736  000137  031056              JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7323  030742                          120$:
7324  030742  004737  045212              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7325  030746  000405                      BR      130$                    ;GO TO 130$ IF NO ERROR
7326  030750  000240                      NOP                     ;RETURN HERE IF ERROR
7327  030752  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7328  030754  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7329  030756  000137  031056              JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7330  030762                          130$:
7331  030762  004737  052120              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7332  030766  000405                      BR      140$                    ;GO TO 140$ IF NO ERROR
7333  030770  000240                      NOP                     ;RETURN HERE IF ERROR
7334  030772  104000                      ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7335  030774  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7336  030776  000137  031056              JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7337  031002                          140$:
7338  031002  004737  061566              JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7339  031006  115760                      .WORD   ND!MSK          ;MASK FOR RMER1
7340  031010  000010                      .WORD   DPE             ;MASK FOR RMER2
7341  031012  000405                      BR      150$                    ;GO TO 150$ IF NO ERROR
7342  031014  000240                      NOP                     ;RETURN HERE IF ERROR
7343  031016  104000                      ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7344  031020  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7345  031022  000137  031056              JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7346  031026                          150$:
```

```
7347  031026  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7348  031032  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
7349  031034  000240                  NOP                     ;RETURN HERE IF ERROR
7350  031036  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7351  031040  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7352  031042  000137  031056          JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
7353  031046                  160$:
7354  031046  005303                  DEC     R3              ;DONE ALL SEEKS??
7355  031050  100402                  BMI     170$            ;YES!!
7356  031052  000137  030676          JMP     90$             ;NO - GO DO NEXT SEEK
7357  031056                  170$:
7358                          ;;*****************************************************************
7359                          ;*TEST 50        SEEK MAXIMUM DIFFERENCE FORWARD
7360
7361                          ;;*****************************************************************
7362  031056                  TST50:
7363  031056  000004                  SCOPE                   ;SCOPE CALL
7364  031060  000240                  NOP                     ;START OF TEST
7365  031062  012737  000002  001206  MOV     #2,$TIMES       ;LOAD ITERATION COUNT
7366  031070  012737  031104  001122  MOV     #1$,$LPADR
7367  031076  012737  031104  001124  MOV     #1$,$LPERR
7368  031104                  1$:
7369  031104  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7370  031110  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7371  031114  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7372  031120  012737  000050  001226  MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7373  031126                  10$:
7374  031126  004737  043216          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7375  031132  054130                  .WORD   054130  ;TASK DESCRIPTOR
7376  031134  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
7377  031136  000240                  NOP                     ;RETURN HERE IF ERROR
7378  031140  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7379  031142  000137  031404          JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7380  031146                  80$:
7381  031146  012737  000000  001432  MOV     #0,RMDC0                ;SEEK TO CYLINDER 0
7382  031154  012737  000000  001404  MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7383  031162  012737  000005  001376  MOV     #SEEK!GO,RMCS10         ;FUNCTION CODE FOR SEEK
7384  031170  012702  001533          MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER TABLE
7385  031174  112722  000006          MOVB    #RMDA,(R2)+             ;WRITE REGISTER INDEX TABLE
7386  031200  112722  000034          MOVB    #RMDC,(R2)+
7387  031204  112722  000000          MOVB    #RMCS1,(R2)+
7388  031210  112722  000200          MOVB    #200,(R2)+              ;TERMINATE THE TABLE
7389  031214                  90$:
7390  031214  004737  044466          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7391  031220  000404                  BR      100$    ;GO TO 100$ IF NO ERROR
7392  031222  000240                  NOP                     ;RETURN HERE IF ERROR
7393  031224  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7394  031226  000137  031404          JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7395  031232                  100$:
7396  031232  004737  044132          JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
7397  031236  004737  045026          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
7398  031242  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7399  031246  000404                  BR      110$    ;GO TO 110$ IF NO ERROR
7400  031250  000240                  NOP                     ;RETURN HERE IF ERROR
7401  031252  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
7402  031254  000137  031404          JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
```

```
7403  031260                          110$:
7404  031260  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7405  031264  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
7406  031266  000240                  NOP                     ;RETURN HERE IF ERROR
7407  031270  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7408  031272  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7409  031274  000137  031404          JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7410  031300                          120$:
7411  031300  004737  052120          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7412  031304  000405                  BR      130$                    ;GO TO 130$ IF NO ERROR
7413  031306  000240                  NOP                     ;RETURN HERE IF ERROR
7414  031310  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7415  031312  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7416  031314  000137  031404          JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7417  031320                          130$:
7418  031320  004737  061566          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7419  031324  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
7420  031326  000010                  .WORD   DPE             ;MASK FOR RMER2
7421  031330  000405                  BR      140$                    ;GO TO 140$ IF NO ERROR
7422  031332  000240                  NOP                     ;RETURN HERE IF ERROR
7423  031334  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7424  031336  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7425  031340  000137  031404          JMP     160$                    ;GO TO 160% IF ERROR WAS FOUND
7426  031344                          140$:
7427  031344  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7428  031350  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
7429  031352  000240                  NOP                     ;RETURN HERE IF ERROR
7430  031354  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7431  031356  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7432  031360  000137  031404          JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7433  031364                          150$:
7434  031364  005737  001432          TST     RMDCO           ;IS TEST DONE??>
7435  031370  001005                  BNE     160$            ;YES!!
7436  031372  012737  001466  001432  MOV     #822.,RMDCO     ;NO - SEEK TO LAST CYLINDER
7437  031400  000137  031214          JMP     90$
7438  031404                          160$:
7439                                  ;;*****************************************************************
7440                                  ;*TEST 51        SEEK ADJACENT FORWARD
7441
7442                                  ;;*****************************************************************
7443  031404                          TST51:
7444  031404  000004                          SCOPE                   ;SCOPE CALL
7445  031406  000240                          NOP                     ;START OF TEST
7446  031410  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7447  031416  012737  031432  001122          MOV     #1$,$LPADR
7448  031424  012737  031432  001124          MOV     #1$,$LPERR
7449  031432                          1$:
7450  031432  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7451  031436  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7452  031442  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7453  031446  012737  000051  001226          MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7454  031454                          10$:
7455  031454  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7456  031460  054130                          .WORD   054130  ;TASK DESCRIPTOR
7457  031462  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
7458  031464  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7459  031466  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7460  031470  000137  031732                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7461  031474                        80$:
7462  031474  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER = 0
7463  031502  012737  000000  001404        MOV     #0,RMDA0        ;TRACK = SECTOR = 0
7464  031510  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
7465  031516  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
7466  031522  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
7467  031526  112722  000006                MOVB    #RMDA,(R2)+
7468  031532  112722  000000                MOVB    #RMCS1,(R2)+
7469  031536  112722  000200                MOVB    #200,(R2)+      ;TERMINATE REGISTER TABLE
7470  031542                        90$:
7471  031542  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7472  031546  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
7473  031550  000240                        NOP                     ;RETURN HERE IF ERROR
7474  031552  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7475  031554  000137  031732                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7476  031560  004737  044132        100$:   JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
7477  031564  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
7478  031570  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7479  031574  000404                        BR      110$    ;GO TO 110$ IF NO ERROR
7480  031576  000240                        NOP                     ;RETURN HERE IF ERROR
7481  031600  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7482  031602  000137  031732                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7483  031606                        110$:
7484  031606  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7485  031612  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
7486  031614  000240                        NOP                     ;RETURN HERE IF ERROR
7487  031616  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7488  031620  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7489  031622  000137  031732                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7490  031626                        120$:
7491  031626  004737  052120                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7492  031632  000405                        BR      130$                    ;GO TO 130$ IF NO ERROR
7493  031634  000240                        NOP                     ;RETURN HERE IF ERROR
7494  031636  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7495  031640  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7496  031642  000137  031732                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7497  031646                        130$:
7498  031646  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7499  031652  115760                        .WORD   ND!MSK          ;MASK FOR RMER1
7500  031654  000010                        .WORD   DPE             ;MASK FOR RMER2
7501  031656  000405                        BR      140$                    ;GO TO 140$ IF NO ERROR
7502  031660  000240                        NOP                     ;RETURN HERE IF ERROR
7503  031662  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7504  031664  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7505  031666  000137  031732                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7506  031672                        140$:
7507  031672  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7508  031676  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7509  031700  000240                        NOP                     ;RETURN HERE IF ERROR
7510  031702  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7511  031704  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7512  031706  000137  031732                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7513  031712                        150$:
7514  031712  005737  001432                TST     RMDC0           ;FIRST TIME THROUGH??
```

```
7515  031716  001005                        BNE     160$            ;NO!!
7516  031720  012737  000001  001432        MOV     #1,RMDCO        ;YES - SEEK TO LAST CYLINDER
7517  031726  000137  031542                JMP     90$
7518  031732                        160$:
7519                                ;;*****************************************************************
7520                                ;*TEST 52        SEEK ADJACENT REVERSE
7521
7522                                ;;*****************************************************************
7523  031732                        TST52:
7524  031732  000004                        SCOPE                   ;SCOPE CALL
7525  031734  000240                        NOP                     ;START OF TEST
7526  031736  012737  000001  001206        MOV     #1,$TIMES       ;LOAD ITERATION COUNT
7527  031744  012737  031760  001122        MOV     #1$,$LPADR
7528  031752  012737  031760  001124        MOV     #1$,$LPERR
7529  031760                        1$:
7530  031760  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7531  031764  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7532  031770  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7533  031774  012737  000052  001226        MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7534  032002                        10$:
7535  032002  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7536  032006  054130                        .WORD   054130  ;TASK DESCRIPTOR
7537  032010  000404                        BR      80$             ;GO TO 80$ IF NO ERROR
7538  032012  000240                        NOP                     ;RETURN HERE IF ERROR
7539  032014  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7540  032016  000137  032260                JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
7541  032022                        80$:
7542  032022  012737  001466  001432        MOV     #822.,RMDCO             ;START AT LAST CYLINDER
7543  032030  012737  000000  001404        MOV     #0,RMDAO                ;TRACK = SECTOR = 0
7544  032036  012737  000005  001376        MOV     #SEEK!GO,RMCS10         ;FUNCTION CODE FOR SEEK
7545  032044  012702  001533                MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER INDEX
7546  032050  112722  000006                MOVB    #RMDA,(R2)+             ;WRITE REGISTER INDEX TABLE
7547  032054  112722  000034                MOVB    #RMDC,(R2)+
7548  032060  112722  000000                MOVB    #RMCS1,(R2)+
7549  032064  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
7550  032070                        90$:
7551  032070  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7552  032074  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
7553  032076  000240                        NOP                     ;RETURN HERE IF ERROR
7554  032100  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7555  032102  000137  032260                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7556  032106  004737  044132        100$:   JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
7557  032112  004737  045026                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
7558  032116  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7559  032122  000404                        BR      110$    ;GO TO 110$ IF NO ERROR
7560  032124  000240                        NOP                     ;RETURN HERE IF ERROR
7561  032126  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7562  032130  000137  032260                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
7563  032134                        110$:
7564  032134  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7565  032140  000405                        BR      120$            ;GO TO 120$ IF NO ERROR
7566  032142  000240                        NOP                     ;RETURN HERE IF ERROR
7567  032144  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7568  032146  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7569  032150  000137  032260                JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
7570  032154                        120$:
```

```
7571  032154  004737  052120            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7572  032160  000405                    BR      130$                    ;GO TO 130$ IF NO ERROR
7573  032162  000240                    NOP                     ;RETURN HERE IF ERROR
7574  032164  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7575  032166  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7576  032170  000137  032260            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7577  032174                    130$:
7578  032174  004737  061566            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7579  032200  115760                    .WORD   ND1MSK          ;MASK FOR RMER1
7580  032202  000010                    .WORD   DPE             ;MASK FOR RMER2
7581  032204  000405                    BR      140$                    ;GO TO 140$ IF NO ERROR
7582  032206  000240                    NOP                     ;RETURN HERE IF ERROR
7583  032210  104000                    ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7584  032212  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7585  032214  000137  032260            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7586  032220                    140$:
7587  032220  004737  046044            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7588  032224  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
7589  032226  000240                    NOP                     ;RETURN HERE IF ERROR
7590  032230  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7591  032232  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7592  032234  000137  032260            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7593  032240                    150$:
7594  032240  022737  001466  001432      CMP     #822.,RMDC0             ;IS THIS THE FIRST SEEK??
7595  032246  001004                      BNE     160$                    ;NO!!
7596  032250  005337  001432              DEC     RMDC0                   ;YES - SEEK TO ADJACENT CYLINDER
7597  032254  000137  032070              JMP     90$                     ;GO SEEK
7598  032260                    160$:
7599                            ;;*****************************************************************
7600                            ;*TEST 53       SEEK INVALID SECTOR
7601                            ;;*****************************************************************
7602  032260                    TST53:
7603  032260  000004                    SCOPE                   ;SCOPE CALL
7604  032262  000240                    NOP                     ;START OF TEST
7605  032264  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7606  032270  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7607  032274  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7608  032300  012737  000053  001226      MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7609
7610  032306                    10$:
7611  032306  004737  043216            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7612  032312  054130                    .WORD   054130  ;TASK DESCRIPTOR
7613  032314  000404                    BR      80$                     ;GO TO 80$ IF NO ERROR
7614  032316  000240                    NOP                     ;RETURN HERE IF ERROR
7615  032320  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7616  032322  000137  032646            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7617  032326  012737  000005  001376  80$:  MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
7618  032334  012737  000000  001430      MOV     #0,RMOF0        ;RESET FORMAT 16 BIT
7619  032342  012737  000000  001432      MOV     #0,RMDC0        ;CYLINDER=0
7620  032350  012737  000036  001404      MOV     #30.,RMDA0      ;SECTOR=30,TRACK=0
7621  032356  012702  001533              MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
7622  032362  112722  000032              MOVB    #RMOF,(R2)+     ;WRITE REGISTER INDEX TABLE
7623  032366  112722  000006              MOVB    #RMDA,(R2)+
7624  032372  112722  000034              MOVB    #RMDC,(R2)+
7625  032376  112722  000000              MOVB    #RMCS1,(R2)+
7626  032402  112722  000200              MOVB    #200,(R2)+      ;TERMINATE TABLE
```

```
7627  032406  004737  044132              JSR     PC,GETSTS       ;SETUP INPUT REGISTER INDEX
7628  032412                      90$:
7629  032412  004737  053360              JSR     PC,CNTCLR
7630  032416  000404                      BR      95$     ;GO TO 95$ IF NO ERROR
7631  032420  000240                      NOP             ;RETURN HERE IF ERROR
7632  032422  104000                      ERROR           ;ERROR NUMBER DEFINED BY SUB
7633  032424  000137  032646              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7634  032430                      95$:
7635  032430  004737  044216              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7636  032434  000404                      BR      100$    ;GO TO 100$ IF NO ERROR
7637  032436  000240                      NOP             ;RETURN HERE IF ERROR
7638  032440  104000                      ERROR           ;ERROR DEFINED BY GET SUB
7639  032442  000137  032646              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7640  032446                      100$:
7641  032446  004737  053476              JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
7642  032452  000405                      BR      110$            ;GO TO 110$ IF NO ERROR
7643  032454  000240                      NOP             ;RETURN HERE IF ERROR
7644  032456  104000                      ERROR           ;ERROR # DEFINED BY CLRSTS SUBROUTINE
7645  032460  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7646  032462  000137  032646              JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
7647  032466                      110$:
7648  032466  004737  044466              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7649  032472  000404                      BR      120$    ;GO TO 120$ IF NO ERROR
7650  032474  000240                      NOP             ;RETURN HERE IF ERROR
7651  032476  104000                      ERROR           ;ERROR DEFINED BY PUT SUB
7652  032500  000137  032626              JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
7653  032504  004737  045026      120$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
7654  032510  004737  044216              JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7655  032514  000404                      BR      125$            ;GO TO 125$ IF NO ERROR
7656  032516  000240                      NOP             ;RETURN HERE IF ERROR
7657  032520  104000                      ERROR           ;ERROR # DEFINED BY GET SUBROUTINE
7658  032522  000137  032646              JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
7659  032526                      125$:
7660  032526  004737  045212              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7661  032532  000405                      BR      130$            ;GO TO 130$ IF NO ERROR
7662  032534  000240                      NOP             ;RETURN HERE IF ERROR
7663  032536  104000                      ERROR           ;ERROR # DEFINED BY PRIERR SUBROUTINE
7664  032540  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7665  032542  000137  032646              JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
7666  032546                      130$:
7667  032546  004737  052120              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7668  032552  000405                      BR      140$            ;GO TO 140$ IF NO ERROR
7669  032554  000240                      NOP             ;RETURN HERE IF ERROR
7670  032556  104000                      ERROR           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7671  032560  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7672  032562  000137  032646              JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
7673  032566                      140$:
7674  032566                      150$:
7675  032566  004737  061566              JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7676  032572  117760                      .WORD   IAE!NDTMSK              ;MASK FOR RMER1
7677  032574  000010                      .WORD   DPE             ;MASK FOR RMER2
7678  032576  000405                      BR      160$            ;GO TO 160$ IF NO ERROR
7679  032600  000240                      NOP             ;RETURN HERE IF ERROR
7680  032602  104000                      ERROR           ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7681  032604  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7682  032606  000137  032626              JMP     170$            ;GO TO 170$ IF ERROR WAS FOUND
```

```
                                160$:
032612                          
032612  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
032616  000403                  BR      170$                    ;GO TO 170$ IF NO ERROR
032620  000240                  NOP                     ;RETURN HERE IF ERROR
032622  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
032624  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
032626  005237  001404  170$:   INC     RMDAO           ;INCREMENT SECTOR ADDRESS
032632  022737  000037  001404  CMP     #31.,RMDAO      ;DONE??
032640  103402                  BLO     180$            ;YES!!
032642  000137  032412          JMP     90$             ;NO - TEST NEXT SECTOR
032646                          180$:

                                ;;*****************************************************************
                                ;*TEST 54        SEEK INVALID TRACK
                                ;;*****************************************************************
032646                          TST54:
032646  000004                  SCOPE                   ;SCOPE CALL
032650  000240                  NOP                     ;START OF TEST
032652  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
032656  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
032662  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
032666  012737  000054  001226  MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

032674  004737  043216          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
032700  054130                  .WORD   054130  ;TASK DESCRIPTOR
032702  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
032704  000240                  NOP                     ;RETURN HERE IF ERROR
032706  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
032710  000137  033262          JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
032714  012737  000005  001376  80$:    MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
032722  012737  000000  001432          MOV     #0,RMDC0        ;DESIRED CYLINDER=0
032730  012737  002400  001404          MOV     #002400,RMDA0   ;SECTOR=0,TRACK=5
032736  012737  000000  001430          MOV     #0,RMOF0        ;18 BIT FORMAT
032744  012702  001533          MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
032750  112722  000032          MOVB    #RMOF,(R2)+     ;WRITE REGISTER INDEX
032754  112722  000034          MOVB    #RMDC,(R2)+     ;TABLE FOR OUTPUT
032760  112722  000006          MOVB    #RMDA,(R2)+
032764  112722  000000          MOVB    #RMCS1,(R2)+
032770  112722  000200          MOVB    #200,(R2)+      ;TERMINATE TABLE
032774  004737  044132          JSR     PC,GETSTS       ;SETUP INPUT TABLE FOR STATUS
033000                          90$:
033000  004737  053360          JSR     PC,CNTCLR
033004  000404                  BR      95$     ;GO TO 95$ IF NO ERROR
033006  000240                  NOP                     ;RETURN HERE IF ERROR
033010  104000                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
033012  000137  033262          JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
033016                          95$:
033016  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
033022  000404                  BR      100$    ;GO TO 100$ IF NO ERROR
033024  000240                  NOP                     ;RETURN HERE IF ERROR
033026  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
033030  000137  033262          JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
033034                          100$:
033034  004737  053476          JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
033040  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
033042  000240                  NOP                     ;RETURN HERE IF ERROR
```

```
 7735  033044  104000                          ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
 7740  033046  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7741  033050  000137  033262                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
 7742  033054                          110$:
 7743  033054  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7744  033060  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
 7745  033062  000240                          NOP                     ;RETURN HERE IF ERROR
 7746  033064  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 7747  033066  000137  033262                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7748  033072  004737  045026          120$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
 7749  033076  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7750  033102  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
 7751  033104  000240                          NOP                     ;RETURN HERE IF ERROR
 7752  033106  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 7753  033110  000137  033262                  JMP     200$    ;GO TO 200$ IF ERROR WAS FOUND
 7754  033114                          130$:
 7755  033114  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7756  033120  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
 7757  033122  000240                          NOP                     ;RETURN HERE IF ERROR
 7758  033124  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7759  033126  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7760  033130  000137  033262                  JMP     200$                    ;GO TO 200$ IF ERROR WAS FOUND
 7761  033134                          140$:
 7762  033134  004737  052120                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 7763  033140  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
 7764  033142  000240                          NOP                     ;RETURN HERE IF ERROR
 7765  033144  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 7766  033146  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7767  033150  000137  033214                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7768  033154                          150$:
 7769  033154                          160$:
 7770  033154  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 7771  033160  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
 7772  033162  000010                          .WORD   DPE             ;MASK FOR RMER2
 7773  033164  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
 7774  033166  000240                          NOP                     ;RETURN HERE IF ERROR
 7775  033170  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 7776  033172  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7777  033174  000137  033214                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7778  033200                          170$:
 7779  033200  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 7780  033204  000403                          BR      180$                    ;GO TO 180$ IF NO ERROR
 7781  033206  000240                          NOP                     ;RETURN HERE IF ERROR
 7782  033210  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 7783  033212  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7784  033214  062737  000400  001404  180$:   ADD     #400,RMDAO      ;INCREMENT TRACK ADDRESS
 7785  033222  022737  003400  001404          CMP     #003400,RMDAO   ;DONE??
 7786  033230  103012                          BHIS    195$            ;NO-DO NEXT TRACK
 7787  033232  032737  010000  001430  190$:   BIT     #FMT16,RMOFO    ;WAS TEST DONE FOR 16 BIT??
 7788  033240  001010                          BNE     200$            ;YES!!
 7789  033242  012737  010000  001430          MOV     #FMT16,RMOFO    ;SET FORMAT 16
 7790  033250  012737  002400  001404          MOV     #002400,RMDAO   ;TRACK = 5
 7791  033256  000137  033000          195$:   JMP     90$             ;DO TEST FOR 16 BIT FORMAT
 7792  033262                          200$:
 7793
 7794                                  ;;*****************************************************************
```

```
                                        ;*TEST 55       SEEK INVALID CYLINDER
                                        ;;*****************************************************************
033262                                  TST55:
033262 000004                           SCOPE                   ;SCOPE CALL
033264 000240                           NOP                     ;START OF TEST
033266 012706 001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
033272 013700 001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
033276 013701 001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
033302 012737 000055 001226             MOV     #55,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
033310 004737 043216                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
033314 054130                           .WORD   054130  ;TASK DESCRIPTOR
033316 000404                           BR      80$             ;GO TO 80$ IF NO ERROR
033320 000240                           NOP                     ;RETURN HERE IF ERROR
033322 104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033324 000137 033674                    JMP     200$            ;GO TO 200$ IF ERROR WAS FOUND
033330 012737 000005 001376     80$:    MOV     #SEEK!GO,RMCS10 ;SEEK FUNCTION CODE
033336 012737 000000 001404             MOV     #0,RMDA0        ;SECTOR = TRACK = 0
033344 012737 001467 001432             MOV     #823.,RMDC0     ;CYLINDER = 823
033352 012737 000000 001430             MOV     #0,RMOF0        ;18 BIT FORMAT
033360 012702 001533                    MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
033364 112722 000006                    MOVB    #RMDA,(R2)+     ;WRITE REGISTER TABLE
033370 112722 000034                    MOVB    #RMDC,(R2)+
033374 112722 000032                    MOVB    #RMOF,(R2)+
033400 112722 000000                    MOVB    #RMCS1,(R2)+
033404 112722 000200                    MOVB    #200,(R2)+      ;TERMINATE TABLE
033410 004737 044132                    JSR     PC,GETSTS       ;SETUP FOR STATUS
033414                          90$:
033414 004737 053360                    JSR     PC,CNTCLR
033420 000404                           BR      95$     ;GO TO 95$ IF NO ERROR
033422 000240                           NOP                     ;RETURN HERE IF ERROR
033424 104000                           ERROR                   ;ERROR NUMBER DEFINED BY SUB
033426 000137 033630                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
033432                          95$:
033432 004737 044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
033436 000404                           BR      100$    ;GO TO 100$ IF NO ERROR
033440 000240                           NOP                     ;RETURN HERE IF ERROR
033442 104000                           ERROR                   ;ERROR DEFINED BY GET SUB
033444 000137 033630                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
033450                          100$:
033450 004737 053476                    JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
033454 000405                           BR      110$            ;GO TO 110$ IF NO ERROR
033456 000240                           NOP                     ;RETURN HERE IF ERROR
033460 104000                           ERROR                   ;ERROR # DEFINED BY CLRSTS SUBROUTINE
033462 004736                           JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033464 000137 033630                    JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
033470                          110$:
033470 004737 044466                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
033474 000404                           BR      120$    ;GO TO 120$ IF NO ERROR
033476 000240                           NOP                     ;RETURN HERE IF ERROR
033500 104000                           ERROR                   ;ERROR DEFINED BY PUT SUB
033502 000137 033630                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
033506 004737 045026            120$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
033512 004737 044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
033516 000404                           BR      130$    ;GO TO 130$ IF NO ERROR
033520 000240                           NOP                     ;RETURN HERE IF ERROR
```

```
7851  033522  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7852  033524  000137  033630                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7853  033530                        130$:
7854  033530  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7855  033534  000405                        BR      140$                    ;GO TO 140$ IF NO ERROR
7856  033536  000240                        NOP                     ;RETURN HERE IF ERROR
7857  033540  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7858  033542  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7859  033544  000137  033630                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7860  033550                        140$:
7861  033550  004737  052120                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
7862  033554  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7863  033556  000240                        NOP                     ;RETURN HERE IF ERROR
7864  033560  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
7865  033562  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7866  033564  000137  033630                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7867  033570                        150$:
7868  033570                        160$:
7869  033570  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7870  033574  117760                        .WORD   IAE!NDTMSK              ;MASK FOR RMER1
7871  033576  000010                        .WORD   DPE             ;MASK FOR RMER2
7872  033600  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
7873  033602  000240                        NOP                     ;RETURN HERE IF ERROR
7874  033604  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7875  033606  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7876  033610  000137  033630                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7877  033614                        170$:
7878  033614  004737  046044                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7879  033620  000403                        BR      180$                    ;GO TO 180$ IF NO ERROR
7880  033622  000240                        NOP                     ;RETURN HERE IF ERROR
7881  033624  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7882  033626  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7883  033630  005237  001432        180$:   INC     RMDCO           ;INCREMENT CYLINDER ADDRESS
7884  033634  022737  002000  001432        CMP     #1024.,RMDCO    ;DONE??
7885  033642  101012                        BHI     195$            ;NO-DO NEXT CYLINDER
7886  033644  032737  010000  001430  190$:  BIT     #FMT16,RMOFO    ;16 BIT FORMAT TESTED??
7887  033652  001010                        BNE     200$            ;YESS!!
7888  033654  012737  010000  001430        MOV     #FMT16,RMOFO    ;SETUP FOR 16 BIT TEST
7889  033662  012737  001467  001432        MOV     #823.,RMDCO
7890  033670  000137  033414          195$: JMP     90$
7891  033674                        200$:
7892
7893                                  ;;*****************************************************************
7894                                  ;*TEST 56        IVC SEEK TEST
7895                                  ;;*****************************************************************
7896  033674                          TST56:
7897  033674  000004                        SCOPE                   ;SCOPE CALL
7898  033676  000240                        NOP                     ;START OF TEST
7899  033700  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7900  033704  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7901  033710  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7902  033714  012737  000056  001226        MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7903
7904  033722  004737  053360                JSR     PC,CNTCLR
7905  033726  000404                        BR      10$     ;GO TO 10$ IF NO ERROR
7906  033730  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
7907  033732  104000                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
7908  033734  000137  034234                JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7909  033740                        10$:
7910  033740  112737  000024  001533        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7911  033746  112737  000200  001534        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7912  033754  012737  000001  001422        MOV     #DMD,RMMR10     ;RMMR1=DMD
7913  033762  004737  044466                JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7914  033766  000404                        BR      20$             ;GO TO 20$ IF NO ERROR
7915  033770  000240                        NOP                     ;RETURN HERE IF ERROR
7916  033772  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7917  033774  000137  034234                JMP     90$             ;GO TO 90$ IF ERROR WAS FOUND
7918  034000                        20$:
7919  034000  012737  000005  001376        MOV     #SEEK!GO,RMCS10
7920  034006  012737  000000  001422        MOV     #0,RMMR10
7921  034014  012737  000000  001404        MOV     #0,RMDA0        ;SECTOR=TRACK=0
7922  034022  012737  000000  001432        MOV     #0,RMDC0        ;CYLINDER=0
7923  034030  012702  001533                MOV     #PUTINX,R2      ;WRITE REGISTER INDEX
7924  034034  112722  000024                MOVB    #RMMR1,(R2)+    ;TABLE
7925  034040  112722  000006                MOVB    #RMDA,(R2)+
7926  034044  112722  000034                MOVB    #RMDC,(R2)+
7927  034050  112722  000000                MOVB    #RMCS1,(R2)+
7928  034054  112722  000200                MOVB    #200,(R2)+
7929  034060  004737  044132                JSR     PC,GETSTS       ;SETUP FOR STATUS
7930  034064  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7931  034070  000404                        BR      30$     ;GO TO 30$ IF NO ERROR
7932  034072  000240                        NOP                     ;RETURN HERE IF ERROR
7933  034074  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7934  034076  000137  034234                JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7935  034102                        30$:
7936  034102  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7937  034106  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
7938  034110  000240                        NOP                     ;RETURN HERE IF ERROR
7939  034112  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7940  034114  000137  034234                JMP     90$     ;GO TO 90$ IF ERROR WAS FOUND
7941  034120                        40$:
7942  034120  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7943  034124  000405                        BR      50$             ;GO TO 50$ IF NO ERROR
7944  034126  000240                        NOP                     ;RETURN HERE IF ERROR
7945  034130  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7946  034132  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7947  034134  000137  034234                JMP     90$             ;GO TO 90$ IF ERROR WAS FOUND
7948  034140  032737  010000  001370  50$:  BIT     #IVC,RMER2I     ;DID INVALID COMMAND SET??
7949  034146  001012                        BNE     60$             ;YES!!
7950  034150  012737  010000  001140        MOV     #IVC,$GDDAT     ;GOOD DATA FOR TYPEOUT
7951  034156  013737  001370  001142        MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
7952  034164  042737  167777  001142        BIC     #^CIVC,$BDDAT
7953  034172  104064                        ERROR   64              ;IVC NOT DETECTED
7954  034174                        60$:
7955  034174                        70$:
7956  034174  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
7957  034200  115760                        .WORD   ND!MSK          ;MASK FOR RMER1
7958  034202  010010                        .WORD   IVC!DPE         ;MASK FOR RMER2
7959  034204  000405                        BR      80$             ;GO TO 80$ IF NO ERROR
7960  034206  000240                        NOP                     ;RETURN HERE IF ERROR
7961  034210  104000                        ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
7962  034212  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
7963  034214  000137  034234            JMP     90$                     ;GO TO 90$ IF ERROR WAS FOUND
7964  034220                    80$:
7965  034220  004737  046044            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7966  034224  000403                    BR      90$                     ;GO TO 90$ IF NO ERROR
7967  034226  000240                    NOP                     ;RETURN HERE IF ERROR
7968  034230  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7969  034232  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7970  034234                    90$:
7971
7972                            ;;**************************************************************
7973                            ;*TEST 57        ABORT SEEK TEST
7974                            ;;**************************************************************
7975  034234                    TST57:
7976  034234  000004                    SCOPE                   ;SCOPE CALL
7977  034236  000240                    NOP                     ;START OF TEST
7978  034240  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7979  034244  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7980  034250  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7981  034254  012737  000057  001226    MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7982
7983  034262  004737  043216            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7984  034266  054130                    .WORD   054130  ;TASK DESCRIPTOR
7985  034270  000404                    BR      80$                     ;GO TO 80$ IF NO ERROR
7986  034272  000240                    NOP                     ;RETURN HERE IF ERROR
7987  034274  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7988  034276  000137  034576            JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
7989  034302  012737  040000  001412  80$:    MOV     #UNS,RMER10     ;SET UNSAFE ERROR
7990  034310  012737  000000  001404            MOV     #0,RMDA0        ;SECTOR = TRACK = 0
```

```
 7991  034316  012737  000000  001432          MOV     #0,RMDC0        ;CYLNDER = 0
 7992  034324  012737  000005  001376          MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
 7993  034332  012702  001533                  MOV     #PUTINX,R2      ;SETUP REGISTER INDEX TABLE
 7994  034336  112722  000006                  MOVB    #RMDA,(R2)+     ;AND OUTPUT BUFFER
 7995  034342  112722  000034                  MOVB    #RMDC,(R2)+
 7996  034346  112722  000014                  MOVB    #RMER1,(R2)+
 7997  034352  112722  000000                  MOVB    #RMCS1,(R2)+
 7998  034356  112722  000200                  MOVB    #200,(R2)+
 7999  034362  004737  044132                  JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
 8000  034366  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 8001  034372  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
 8002  034374  000240                          NOP                     ;RETURN HERE IF ERROR
 8003  034376  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 8004  034400  000137  034576                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8005  034404                          90$:
 8006  034404  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8007  034410  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
 8008  034412  000240                          NOP                     ;RETURN HERE IF ERROR
 8009  034414  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 8010  034416  000137  034576                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8011  034422  032737  020000  001340  100$:   BIT     #PIP,RMDSI      ;DID DRIVE START SEEK??
 8012  034430  001411                          BEQ     110$            ;NO!!
 8013  034432  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
 8014  034440  042737  157777  001142          BIC     #^CPIP,$BDDAT
 8015  034446  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
 8016  034452  104065                          ERROR   65              ;DRIVE SHOULD NOT SEEK
 8017  034454  004737  045026          110$:   JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
 8018  034460  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 8019  034464  000404                          BR      120$    ;GO TO 120$ IF NO ERROR
 8020  034466  000240                          NOP                     ;RETURN HERE IF ERROR
 8021  034470  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 8022  034472  000137  034576                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
 8023  034476                          120$:
 8024  034476  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8025  034502  000405                          BR      130$                    ;GO TO 130$ IF NO ERROR
 8026  034504  000240                          NOP                     ;RETURN HERE IF ERROR
 8027  034506  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8028  034510  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8029  034512  000137  034576                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8030  034516                          130$:
 8031  034516  004737  061062                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
 8032  034522  000405                          BR      140$                    ;GO TO 140$ IF NO ERROR
 8033  034524  000240                          NOP                     ;RETURN HERE IF ERROR
 8034  034526  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
 8035  034530  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8036  034532  000137  034576                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8037  034536                          140$:
 8038  034536  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 8039  034542  155760                          .WORD   NDTMSK!UNS              ;MASK FOR RMER1
 8040  034544  000010                          .WORD   DPE             ;MASK FOR RMER2
 8041  034546  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
 8042  034550  000240                          NOP                     ;RETURN HERE IF ERROR
 8043  034552  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 8044  034554  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8045  034556  000137  034576                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
 8046  034562                          150$:
```

```
8047  034562  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8048  034566  000403                  BR      160$                    ;GO TO 160$ IF NO ERROR
8049  034570  000240                  NOP                     ;RETURN HERE IF ERROR
8050  034572  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8051  034574  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8052  034576                  160$:
8053
8054                          ;;*****************************************************************
8055                          ;*TEST 60        SEEK AT OFFSET
8056                          ;;*****************************************************************
8057  034576                  TST60:
8058  034576  000004                  SCOPE                   ;SCOPE CALL
8059  034600  000240                  NOP                     ;START OF TEST
8060  034602  012737  000001  001206  MOV     #1,$TIMES       ;LOAD ITERATION COUNT
8061  034610  012737  034624  001122  MOV     #1$,$LPADR
8062  034616  012737  034624  001124  MOV     #1$,$LPERR
8063  034624                  1$:
8064  034624  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8065  034630  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8066  034634  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8067  034640  012737  000060  001226  MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8068  034646                  10$:
8069  034646  004737  043216          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8070  034652  054130                  .WORD   054130  ;TASK DESCRIPTOR
8071  034654  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
8072  034656  000240                  NOP                     ;RETURN HERE IF ERROR
8073  034660  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8074  034662  000137  035144          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
8075  034666                  20$:
8076  034666  012737  000000  001404  MOV     #0,RMDA0        ;TRACK = SECTOR = 0
8077  034674  012737  000000  001432  MOV     #0,RMDC0        ;CYLINDER = 0
8078  034702  012737  000015  001376  MOV     #OFFSET!GO,RMCS10       ;LOAD OFFSET COMMAND IN BUFFER
8079  034710  012702  001533          MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
8080  034714  112722  000034          MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
8081  034720  112722  000006          MOVB    #RMDA,(R2)+
8082  034724  112722  000000          MOVB    #RMCS1,(R2)+
8083  034730  112722  000200          MOVB    #200,(R2)+      ;WRITE TERMINATOR
8084  034734  004737  044466          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8085  034740  000404                  BR      30$     ;GO TO 30$ IF NO ERROR
8086  034742  000240                  NOP                     ;RETURN HERE IF ERROR
8087  034744  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
8088  034746  000137  035144          JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8089  034752  004737  044132  30$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
8090  034756  004737  045026          JSR     PC,TIMOUT       ;WAIT FOR OFFSET TO COMPLETE
8091  034762                  40$:
8092  034762  012737  000005  001376  MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
8093  034770  112737  000000  001533  MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
8094  034776  112737  000200  001534  MOVB    #200,PUTINX+1
8095  035004  004737  044466          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8096  035010  000404                  BR      50$     ;GO TO 50$ IF NO ERROR
8097  035012  000240                  NOP                     ;RETURN HERE IF ERROR
8098  035014  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
8099  035016  000137  035144          JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8100  035022                  50$:
8101  035022  004737  045026          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8102  035026                  60$:
```

```
8103  035026  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8104  035032  000404                  BR      100$    ;GO TO 100$ IF NO ERROR
8105  035034  000240                  NOP                     ;RETURN HERE IF ERROR
8106  035036  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
8107  035040  000137  035144          JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
8108  035044                  100$:
8109  035044  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8110  035050  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
8111  035052  000240                  NOP                     ;RETURN HERE IF ERROR
8112  035054  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8113  035056  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8114  035060  000137  035144          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
8115  035064                  110$:
8116  035064  004737  052120          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
8117  035070  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
8118  035072  000240                  NOP                     ;RETURN HERE IF ERROR
8119  035074  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
8120  035076  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8121  035100  000137  035144          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
8122  035104                  120$:
8123  035104  004737  061566          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8124  035110  115760                  .WORD   ND!MSK          ;MASK FOR RMER1
8125  035112  000010                  .WORD   DPE             ;MASK FOR RMER2
8126  035114  000405                  BR      130$                    ;GO TO 130$ IF NO ERROR
8127  035116  000240                  NOP                     ;RETURN HERE IF ERROR
8128  035120  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8129  035122  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8130  035124  000137  035144          JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
8131  035130                  130$:
8132  035130  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8133  035134  000403                  BR      140$                    ;GO TO 140$ IF NO ERROR
8134  035136  000240                  NOP                     ;RETURN HERE IF ERROR
8135  035140  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8136  035142  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8137  035144                  140$:
8138                          ;;*****************************************************************
8139                          ;*TEST 61        LOOK AHEAD TEST
8140                          ;;*****************************************************************
8141  035144                  TST61:
8142  035144  000004                  SCOPE                   ;SCOPE CALL
8143  035146  000240                  NOP                     ;START OF TEST
8144  035150  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8145  035154  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8146  035160  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8147  035164  012737  000061  001226  MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8148
8149  035172  004737  053360          JSR     PC,CNTCLR
8150  035176  000404                  BR      10$     ;GO TO 10$ IF NO ERROR
8151  035200  000240                  NOP                     ;RETURN HERE IF ERROR
8152  035202  104000                  ERROR                   ;ERROR NUMBER DEFINED BY SUB
8153  035204  000137  035616          JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8154  035210                  10$:
8155  035210  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
8156  035214  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
8157  035220  012737  035604  000004  MOV     #145$,ERRVEC    ;SETUP FOR BUS TIMOUT
8158  035226  012737  000300  000006  MOV     #PR6,ERRVEC+2
```

```
8159  035234  012703  000041                    MOV     #33.,R3          ;R3=SAMPLE COUNT
8160  035240  012737  010000  001430            MOV     #FMT16,RMOFO     ;SETUP 16 BIT MODE
8161  035246  012737  000000  001404            MOV     #0,RMDAO         ;SEARCH SECTOR 0
8162  035254  012737  000000  001432            MOV     #0,RMDCO         ;CYLINDER 0
8163  035262  012737  000031  001376            MOV     #SEARCH!GO,RMCS1O        ;RECALIBRATE COMMAND
8164  035270  012702  001533                    MOV     #PUTINX,R2
8165  035274  112722  000006                    MOVB    #RMDA,(R2)+
8166  035300  112722  000034                    MOVB    #RMDC,(R2)+
8167  035304  112722  000032                    MOVB    #RMOF,(R2)+
8168  035310  112722  000000                    MOVB    #RMCS1,(R2)+
8169  035314  112722  000200                    MOVB    #200,(R2)+
8170  035320  004737  044132                    JSR     PC,GETSTS
8171  035324                          20$:
8172  035324  004737  044466                    JSR     PC,PUT           ;GO WRITE REGISTERS VIA PUT SUB
8173  035330  000404                            BR      30$      ;GO TO 30$ IF NO ERROR
8174  035332  000240                            NOP                      ;RETURN HERE IF ERROR
8175  035334  104000                            ERROR                    ;ERROR DEFINED BY PUT SUB
8176  035336  000137  035606                    JMP     150$     ;GO TO 150$ IF ERROR WAS FOUND
8177  035342  004737  045026          30$:     JSR     PC,TIMOUT        ;WAIT FOR COMPLETION
8178  035346  004737  044216                    JSR     PC,GET           ;GO READ REGISTERS VIA GET SUB
8179  035352  000404                            BR      40$      ;GO TO 40$ IF NO ERROR
8180  035354  000240                            NOP                      ;RETURN HERE IF ERROR
8181  035356  104000                            ERROR                    ;ERROR DEFINED BY GET SUB
8182  035360  000137  035606                    JMP     150$     ;GO TO 150$ IF ERROR WAS FOUND
8183  035364                          40$:
8184  035364  004737  045212                    JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
8185  035370  000405                            BR      50$                      ;GO TO 50$ IF NO ERROR
8186  035372  000240                            NOP                      ;RETURN HERE IF ERROR
8187  035374  104000                            ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
8188  035376  004736                            JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
8189  035400  000137  035606                    JMP     150$                     ;GO TO 150$ IF ERROR WAS FOUND
8190  035404                          50$:
8191  035404  012704  106126          60$:     MOV     #BUFFER,R4       ;R4=STORAGE ADDRESS
8192  035410  012705  177777                    MOV     #-1,R5                   ;R5=GROSS TIMER
8193  035414  016014  000020          70$:     MOV     RMLA(RO),(R4)    ;STORE RMLA SAMPLE
8194  035420  016002  000020                    MOV     RMLA(RO),R2      ;GET ANOTHER LOOK
8195  035424  020214                            CMP     R2,(R4)          ;ARE SAMPLES SAME??
8196  035426  001403                            BEQ     80$              ;YES!!
8197  035430  005305                            DEC     R5               ;TIMEOUT??
8198  035432  001370                            BNE     70$              ;NO - TRY AGAIN
8199  035434  000411                            BR      100$             ;PROGRAM TIMEOUT
8200  035436  062704  000002          80$:     ADD     #2,R4            ;ADVANCE STORAGE ADDRESS
8201  035442  005303                            DEC     R3               ;ALL SAMPLES TAKEN??
8202  035444  001410                            BEQ     110$             ;YES!!
8203  035446  026002  000020          90$:     CMP     RMLA(RO),R2      ;WAIT FOR CHANGE
8204  035452  001360                            BNE     70$              ;YES!!
8205  035454  005305                            DEC     R5               ;TIMEOUT??
8206  035456  001373                            BNE     90$              ;NO
8207  035460  104352                  100$:     ERROR   352              ;PROGRAM TIMEOUT
8208  035462  000137  035616                    JMP     160$
8209  035466  012703  000040          110$:    MOV     #32.,R3          ;R3 = NUMBER OF COMPARES
8210  035472  012702  003700                    MOV     #3700,R2                 ;R2 = MAXIMUM SAMPLE
8211  035476  012704  106126                    MOV     #BUFFER,R4       ;R4 = BUFFER ADDRESS
8212  035502  032737  010000  001430            BIT     #FMT16,RMOFO     ;WAS SAMPLE FOR 16 BIT MODE??
8213  035510  001004                            BNE     120$             ;YES
8214  035512  012703  000036                    MOV     #30.,R3          ;CHANGE COMPARE COUNT AND
```

```
8215  035516  012702  003500              MOV     #3500,R2                ;MAXIMUM SAMPLE
8216  035522  012405              120$:   MOV     (R4)+,R5                ;GET A SAMPLE AND INCREMENT
8217  035524  062705  000100              ADD     #100,R5                 ;FOR EXPECTED VALUE OF NEXT
8218  035530  020205                      CMP     R2,R5                   ;SHOULD NEXT BE 0??
8219  035532  103001                      BHIS    130$                    ;NO!!
8220  035534  005005                      CLR     R5                      ;YES - CHANGE EXPECTED
8221  035536  020514              130$:   CMP     R5,(R4)                 ;IS NEXT SAMPLE CORRECT??
8222  035540  001406                      BEQ     140$                    ;YES!!
8223  035542  010537  001140              MOV     R5,$GDDAT               ;EXPECTED VALUE
8224  035546  011437  001142              MOV     (R4),$BDDAT             ;RECEIVED VALUE
8225  035552  104353                      ERROR   353                     ;FAILURE IN LOOK AHEAD TEST
8226  035554  000414                      BR      150$
8227  035556  005303              140$:   DEC     R3                      ;ALL SAMPLES CHECKED??
8228  035560  001360                      BNE     120$                    ;NO!!
8229  035562  032737  010000  001430      BIT     #FMT16,RMOFO            ;DONE TEST??
8230  035570  001406                      BEQ     150$                    ;YES!!
8231  035572  005037  001430              CLR     RMOFO                   ;SETUP FOR 18 BIT MODE
8232  035576  012703  000037              MOV     #31.,R3
8233  035602  000650                      BR      20$                     ;DO AGAIN FOR 18 BIT MODE
8234  035604  022626              145$:   CMP     (SP)+,(SP)+             ;RESTORE STACK
8235  035606                      150$:
8236  035606  012637  000006              MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
8237  035612  012637  000004              MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
8238  035616                      160$:
8239                              ;;**************************************************************
8240                              ;*TEST 62        SEARCH ON CYLINDER
8241                              ;;**************************************************************
8242  035616                      TST62:
8243  035616  000004                      SCOPE                           ;SCOPE CALL
8244  035620  000240                      NOP                             ;START OF TEST
8245  035622  012706  001100              MOV     #STACK,SP               ;INITIALIZE STACK POINTER
8246  035626  013700  001276              MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
8247  035632  013701  001446              MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
8248  035636  012737  000062  001226      MOV     #62,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
8249  035644  004737  043216              JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8250  035650  054130                      .WORD   054130  ;TASK DESCRIPTOR
8251  035652  000404                      BR      80$                             ;GO TO 80$ IF NO ERROR
8252  035654  000240                      NOP                             ;RETURN HERE IF ERROR
8253  035656  104000                      ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8254  035660  000137  036274              JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
8255  035664  012703  000035      80$:    MOV     #29.,R3                 ;R3=MAXIMUM SECTOR ADDRESS
8256  035670  012737  000000  001404      MOV     #0,RMDAO                ;TRACK=SECTOR=0
8257  035676  012737  000000  001432      MOV     #0,RMDCO                ;CYLINDER=0
8258  035704  012737  000000  001430      MOV     #0,RMOFO                ;18 BIT FORMAT
8259  035712  012737  000005  001376  85$:  MOV     #SEEK!GO,RMCS10         ;SEEK COMMAND
8260  035720  012702  001533              MOV     #PUTINX,R2              ;SETUP REGISTER INDEX TABLE
8261  035724  112722  000006              MOVB    #RMDA,(R2)+             ;FOR SEEK TO CYLINDER 0
8262  035730  112722  000034              MOVB    #RMDC,(R2)+
8263  035734  112722  000032              MOVB    #RMOF,(R2)+
8264  035740  112722  000000              MOVB    #RMCS1,(R2)+
8265  035744  112722  000200              MOVB    #200,(R2)+
8266  035750  004737  044132              JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
8267  035754  004737  044466              JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
8268  035760  000404                      BR      90$     ;GO TO 90$ IF NO ERROR
8269  035762  000240                      NOP                             ;RETURN HERE IF ERROR
8270  035764  104000                      ERROR                           ;ERROR DEFINED BY PUT SUB
```

```
8271  035766  000137  036274          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8272  035772  004737  045026  90$:    JSR     PC,TIMOUT
8273  035776  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8274  036002  000404                  BR      100$    ;GO TO 100$ IF NO ERROR
8275  036004  000240                  NOP                     ;RETURN HERE IF ERROR
8276  036006  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
8277  036010  000137  036274          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8278  036014                  100$:
8279  036014  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8280  036020  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
8281  036022  000240                  NOP                     ;RETURN HERE IF ERROR
8282  036024  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8283  036026  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8284  036030  000137  036274          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8285  036034                  110$:
8286  036034  004737  052120          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
8287  036040  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
8288  036042  000240                  NOP                     ;RETURN HERE IF ERROR
8289  036044  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
8290  036046  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8291  036050  000137  036274          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8292  036054  012737  000031  001376  120$:   MOV     #SEARCH!GO,RMCS1O;STORE SEARCH COMMAND
8293  036062  004737  044466          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8294  036066  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
8295  036070  000240                  NOP                     ;RETURN HERE IF ERROR
8296  036072  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
8297  036074  000137  036274          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8298  036100                  130$:
8299  036100  004737  045026          JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
8300  036104  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8301  036110  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
8302  036112  000240                  NOP                     ;RETURN HERE IF ERROR
8303  036114  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
8304  036116  000137  036274          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8305  036122                  140$:
8306  036122  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8307  036126  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
8308  036130  000240                  NOP                     ;RETURN HERE IF ERROR
8309  036132  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8310  036134  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8311  036136  000137  036274          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8312  036142                  150$:
8313  036142  004737  057516          JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
8314  036146  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
8315  036150  000240                  NOP                     ;RETURN HERE IF ERROR
8316  036152  104000                  ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
8317  036154  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8318  036156  000137  036274          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8319  036162                  160$:
8320  036162  004737  061566          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8321  036166  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
8322  036170  000010                  .WORD   DPE             ;MASK FOR RMER2
8323  036172  000405                  BR      170$                    ;GO TO 170$ IF NO ERROR
8324  036174  000240                  NOP                     ;RETURN HERE IF ERROR
8325  036176  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8326  036200  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
8327  036202  000137  036274            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8328  036206                    170$:
8329  036206  004737  046044            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8330  036212  000405                    BR      180$                    ;GO TO 180$ IF NO ERROR
8331  036214  000240                    NOP                     ;RETURN HERE IF ERROR
8332  036216  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8333  036220  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8334  036222  000137  036274            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8335  036226  005237  001404    180$:   INC     RMDAO           ;INCREMENT SECTOR ADDRESS
8336  036232  020337  001404            CMP     R3,RMDAO        ;DONE ALL SECTORS??
8337  036236  103306                    BHIS    120$            ;NO!!
8338  036240  032737  010000  001430    BIT     #FMT16,RMOFO    ;IS 16 BIT FORMAT DONE??
8339  036246  001012                    BNE     190$            ;YES!!
8340  036250  012737  010000  001430    MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
8341  036256  012703  000037            MOV     #31.,R3         ;R3=MAXIMUM SECTOR ADDRESS
8342  036262  012737  000000  001404    MOV     #0,RMDAO        ;START AT SECTOR 0
8343  036270  000137  035712            JMP     85$             ;GO TEST 16 BIT MODE
8344  036274                    190$:
8345
8346
8347
8348                              ;;*****************************************************************
8349                              ;*TEST 63       SEARCH OFF CYLINDER
8350                              ;;*****************************************************************
8351  036274                    TST63:
8352  036274  000004                    SCOPE                   ;SCOPE CALL
8353  036276  000240                    NOP                     ;START OF TEST
8354  036300  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8355  036304  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8356  036310  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8357  036314  012737  000063  001226    MOV     #63,$TESTN      ;SET TEST NUMBER IN APT MAIL BOX
8358  036322  004737  043216            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8359  036326  054130                    .WORD   054130  ;TASK DESCRIPTOR
8360  036330  000404                    BR      80$                     ;GO TO 80$ IF NO ERROR
8361  036332  000240                    NOP                     ;RETURN HERE IF ERROR
8362  036334  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8363  036336  000137  037002            JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
8364  036342  012737  000000  001404  80$:   MOV     #0,RMDAO
8365  036350  012737  000000  001430        MOV     #0,RMOFO        ;START WITH 18 BIT MODE
8366  036356  012704  000035            MOV     #29.,R4         ;R4=MAXIMUM SECTOR ADDRESS
8367  036362  012703  000633            MOV     #411.,R3        ;R3=SEARCH CYLINDER
8368  036366  012705  000634            MOV     #412.,R5        ;R5=SEEK CYLINDER
8369  036372  012737  000005  001376  90$:   MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
8370  036400  010537  001432            MOV     R5,RMDCO        ;LOAD CYLINDER ADDRESS
8371  036404  012702  001533            MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
8372  036410  112722  000032            MOVB    #RMOF,(R2)+     ;SETUP REGISTER INDEX TABLE
8373  036414  112722  000034            MOVB    #RMDC,(R2)+     ;FOR SEEK COMMAND
8374  036420  112722  000006            MOVB    #RMDA,(R2)+
8375  036424  112722  000000            MOVB    #RMCS1,(R2)+
8376  036430  112722  000200            MOVB    #200,(R2)+      ;TERMINATE TABLE
8377  036434  004737  044132            JSR     PC,GETSTS       ;SETUP FOR STATUS
8378  036440  004737  044466            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8379  036444  000404                    BR      100$    ;GO TO 100$ IF NO ERROR
8380  036446  000240                    NOP                     ;RETURN HERE IF ERROR
8381  036450  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
8382  036452  000137  037002            JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
```

```
8383  036456  004737  045026          100$:  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
8384  036462  004737  044216                 JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8385  036466  000404                         BR      110$    ;GO TO 110$ IF NO ERROR
8386  036470  000240                         NOP                     ;RETURN HERE IF ERROR
8387  036472  104000                         ERROR                   ;ERROR DEFINED BY GET SUB
8388  036474  000137  037002                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
8389  036500                         110$:
8390  036500  004737  045212                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8391  036504  000405                         BR      120$                    ;GO TO 120$ IF NO ERROR
8392  036506  000240                         NOP                     ;RETURN HERE IF ERROR
8393  036510  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8394  036512  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8395  036514  000137  037002                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
8396  036520                         120$:
8397  036520  004737  052120                 JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
8398  036524  000405                         BR      130$                    ;GO TO 130$ IF NO ERROR
8399  036526  000240                         NOP                     ;RETURN HERE IF ERROR
8400  036530  104000                         ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
8401  036532  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8402  036534  000137  037002                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
8403  036540  010337  001432          130$:  MOV     R3,RMDCO        ;LOAD CYLINDER ADDRESS
8404  036544  012737  000031  001376         MOV     #SEARCH!GO,RMCS1;LOAD SEARCH COMMAND
8405  036552  004737  044466                 JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8406  036556  000404                         BR      140$    ;GO TO 140$ IF NO ERROR
8407  036560  000240                         NOP                     ;RETURN HERE IF ERROR
8408  036562  104000                         ERROR                   ;ERROR DEFINED BY PUT SUB
8409  036564  000137  037002                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
8410  036570  004737  045026          140$:  JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
8411  036574  004737  044216                 JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8412  036600  000404                         BR      150$    ;GO TO 150$ IF NO ERROR
8413  036602  000240                         NOP                     ;RETURN HERE IF ERROR
8414  036604  104000                         ERROR                   ;ERROR DEFINED BY GET SUB
8415  036606  000137  037002                 JMP     230$    ;GO TO 230$ IF ERROR WAS FOUND
8416  036612                         150$:
8417  036612  004737  045212                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8418  036616  000405                         BR      160$                    ;GO TO 160$ IF NO ERROR
8419  036620  000240                         NOP                     ;RETURN HERE IF ERROR
8420  036622  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8421  036624  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8422  036626  000137  037002                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
8423  036632                         160$:
8424  036632  004737  057516                 JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
8425  036636  000405                         BR      170$                    ;GO TO 170$ IF NO ERROR
8426  036640  000240                         NOP                     ;RETURN HERE IF ERROR
8427  036642  104000                         ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
8428  036644  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8429  036646  000137  037002                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
8430  036652                         170$:
8431  036652  004737  061566                 JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8432  036656  115760                         .WORD   ND!MSK          ;MASK FOR RMER1
8433  036660  000010                         .WORD   DPE             ;MASK FOR RMER2
8434  036662  000405                         BR      180$                    ;GO TO 180$ IF NO ERROR
8435  036664  000240                         NOP                     ;RETURN HERE IF ERROR
8436  036666  104000                         ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8437  036670  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8438  036672  000137  037002                 JMP     230$                    ;GO TO 230$ IF ERROR WAS FOUND
```

```
8439  036676                          180$:
8440  036676  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8441  036702  000405                          BR      190$                    ;GO TO 190$ IF NO ERROR
8442  036704  000240                          NOP                     ;RETURN HERE IF ERROR
8443  036706  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8444  036710  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8445  036712  000137  036716                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8446  036716  005303                  190$:   DEC     R3              ;DECREMENT SEARCH CYLINDER
8447  036720  005205                          INC     R5              ;INCREMENT SEEK CYLINDER
8448  036722  005237  001404                  INC     RMDAO           ;INCREMENT SECTOR ADDRESS
8449  036726  020437  001404                  CMP     R4,RMDAO        ;DONE ALL SECTORS??
8450  036732  103401                          BLO     210$            ;YES!!
8451  036734  000616                          BR      90$             ;DO NEXT SECTOR
8452  036736  032737  010000  001430  210$:   BIT     #FMT16,RMOFO    ;DONE BOTH FORMATS??
8453  036744  001016                          BNE     230$            ;YES!!
8454  036746  012704  000037                  MOV     #31.,R4         ;R4=MAXIMUM SECTOR ADDRESS
8455  036752  012703  000634                  MOV     #412.,R3        ;R3=SEARCH CYLINDER
8456  036756  012705  000633                  MOV     #411.,R5        ;R5=SEEK CYLINDER
8457  036762  012737  010000  001430          MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
8458  036770  012737  000000  001404          MOV     #0,RMDAO        ;START AT SECTOR 0
8459  036776  000137  036372                  JMP     90$             ;START 16 BIT TEST
8460  037002                          230$:
8461                                  ;;*****************************************************************
8462                                  ;*TEST 64        SEARCH INVALID SECTOR
8463                                  ;;*****************************************************************
8464  037002                          TST64:
8465  037002  000004                          SCOPE                   ;SCOPE CALL
8466  037004  000240                          NOP                     ;START OF TEST
8467  037006  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8468  037012  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8469  037016  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8470  037022  012737  000064  001226          MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8471
8472  037030  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8473  037034  054130                          .WORD   054130  ;TASK DESCRIPTOR
8474  037036  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8475  037040  000240                          NOP                     ;RETURN HERE IF ERROR
8476  037042  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8477  037044  000137  037362                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8478  037050  012737  000000  001430  80$:    MOV     #0,RMOFO        ;SET 18 BIT FORMAT
8479  037056  012737  000036  001404          MOV     #30.,RMDAO      ;START WITH SECTOR 30
8480  037064  012737  000000  001432          MOV     #0,RMDCO        ;CYLINDER=0
8481  037072  012737  000031  001376          MOV     #SEARCH!GO,RMCS1;SEARCH COMMAND
8482  037100  012702  001533                  MOV     #PUTINX,R2      ;WRITE REGISTER TABLE FOR
8483  037104  112722  000034                  MOVB    #RMDC,(R2)+     ;COMMAND
8484  037110  112722  000032                  MOVB    #RMOF,(R2)+
8485  037114  112722  000006                  MOVB    #RMDA,(R2)+
8486  037120  112722  000000                  MOVB    #RMCS1,(R2)+
8487  037124  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
8488  037130  004737  044132                  JSR     PC,GETSTS       ;SETUP STATUS FETCH
8489  037134                          90$:
8490  037134  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8491  037140  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
8492  037142  000240                          NOP                     ;RETURN HERE IF ERROR
8493  037144  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
8494  037146  000137  037330                  JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
```

```
8495  037152  004737  045026          100$:  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8496  037156  004737  044216                 JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8497  037162  000404                         BR      120$    ;GO TO 120$ IF NO ERROR
8498  037164  000240                         NOP                     ;RETURN HERE IF ERROR
8499  037166  104000                         ERROR                   ;ERROR DEFINED BY GET SUB
8500  037170  000137  037330                 JMP     170$    ;GO TO 170$ IF ERROR WAS FOUND
8501  037174                          120$:
8502  037174  004737  045212                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8503  037200  000405                         BR      130$                    ;GO TO 130$ IF NO ERROR
8504  037202  000240                         NOP                     ;RETURN HERE IF ERROR
8505  037204  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8506  037206  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8507  037210  000137  037330                 JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8508  037214  032737  002000  001342  130$:  BIT     #IAE,RMER1I     ;DID "IAE" SET??
8509  037222  001012                         BNE     140$            ;YES!!
8510  037224  012737  002000  001140         MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
8511  037232  013737  001342  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8512  037240  042737  175777  001142         BIC     #^CIAE,$BDDAT
8513  037246  104257                         ERROR   257             ;IVC NOT DETECTED
8514  037250                          140$:
8515  037250  004737  061062                 JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
8516  037254  000405                         BR      150$                    ;GO TO 150$ IF NO ERROR
8517  037256  000240                         NOP                     ;RETURN HERE IF ERROR
8518  037260  104000                         ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
8519  037262  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8520  037264  000137  037330                 JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8521  037270                          150$:
8522  037270  004737  061566                 JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8523  037274  117760                         .WORD   ND!MSK!IAE              ;MASK FOR RMER1
8524  037276  000010                         .WORD   DPE             ;MASK FOR RMER2
8525  037300  000405                         BR      160$                    ;GO TO 160$ IF NO ERROR
8526  037302  000240                         NOP                     ;RETURN HERE IF ERROR
8527  037304  104000                         ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8528  037306  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8529  037310  000137  037330                 JMP     170$                    ;GO TO 170$ IF ERROR WAS FOUND
8530  037314                          160$:
8531  037314  004737  046044                 JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8532  037320  000403                         BR      170$                    ;GO TO 170$ IF NO ERROR
8533  037322  000240                         NOP                     ;RETURN HERE IF ERROR
8534  037324  104000                         ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8535  037326  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8536  037330  005237  001404          170$:  INC     RMDAO           ;INCREMENT SECTOR ADDRESS
8537  037334  022737  000037  001404         CMP     #31.,RMDAO      ;DONE??
8538  037342  103407                         BLO     190$            ;YES!!
8539  037344  004737  053360                 JSR     PC,CNTCLR
8540  037350  000402                         BR      180$    ;GO TO 180$ IF NO ERROR
8541  037352  000240                         NOP                     ;RETURN HERE IF ERROR
8542  037354  104000                         ERROR                   ;ERROR NUMBER DEFINED BY SUB
8543  037356  000137  037134          180$:  JMP     90$             ;TEST NEXT SECTOR
8544  037362                          190$:
8545
8546
8547
8548
8549
8550
```

```
8551
8552
8553
8554
8555
8556
8557                                    ;;*****************************************************************
8558                                    ;*TEST 65        SEARCH INVALID TRACK
8559                                    ;;*****************************************************************
8560   037362                           TST65:
8561   037362  000004                            SCOPE                   ;SCOPE CALL
8562   037364  000240                            NOP                     ;START OF TEST
8563   037366  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8564   037372  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8565   037376  013701  001446                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8566   037402  012737  000065  001226            MOV     #65,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8567   037410  004737  043216                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8568   037414  054130                            .WORD   054130  ;TASK DESCRIPTOR
8569   037416  000404                            BR      80$             ;GO TO 80$ IF NO ERROR
8570   037420  000240                            NOP                     ;RETURN HERE IF ERROR
8571   037422  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8572   037424  000137  037750                    JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
8573
8574   037430  012737  000000  001430  80$:     MOV     #0,RMOF0        ;SET 18 BIT FORMAT
8575   037436  012737  002400  001404            MOV     #002400,RMDA0   ;START WITH TRACK 5, SECTOR 0
8576   037444  012737  000000  001432            MOV     #0,RMDC0        ;CYLINDER=0
8577   037452  012737  000031  001376            MOV     #SEARCH!GO,RMCS10;SEARCH COMMAND
8578   037460  012702  001533                    MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
8579   037464  112722  000006                    MOVB    #RMDA,(R2)+
8580   037470  112722  000034                    MOVB    #RMDC,(R2)+
8581   037474  112722  000032                    MOVB    #RMOF,(R2)+
8582   037500  112722  000000                    MOVB    #RMCS1,(R2)+
8583   037504  112722  000200                    MOVB    #200,(R2)+      ;TERMINATE TABLE
8584   037510  004737  044132                    JSR     PC,GETSTS
8585   037514                           90$:
8586   037514  004737  044466                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8587   037520  000404                            BR      100$    ;GO TO 100$ IF NO ERROR
8588   037522  000240                            NOP                     ;RETURN HERE IF ERROR
8589   037524  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
8590   037526  000137  037670                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8591   037532  004737  045026           100$:    JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8592   037536  004737  044216                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8593   037542  000404                            BR      110$    ;GO TO 110$ IF NO ERROR
8594   037544  000240                            NOP                     ;RETURN HERE IF ERROR
8595   037546  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
8596   037550  000137  037670                    JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8597   037554                           110$:
8598   037554  004737  045212                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8599   037560  000405                            BR      120$            ;GO TO 120$ IF NO ERROR
8600   037562  000240                            NOP                     ;RETURN HERE IF ERROR
8601   037564  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8602   037566  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8603   037570  000137  037670                    JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8604   037574  032737  002000  001342  120$:    BIT     #IAE,RMER1I     ;DID "IAE" SET??
8605   037602  001012                            BNE     130$            ;YES!!
8606   037604  012737  002000  001140            MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
```

```
8607  037612  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8608  037620  042737  175777  001142          BIC     #↑CIAE,$BDDAT
8609  037626  104257                          ERROR   257             ;IAE NOT SET DURING SEARCH
8610  037630                          130$:
8611  037630                          140$:
8612  037630  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8613  037634  117760                          .WORD   ND↑MSK!IAE              ;MASK FOR RMER1
8614  037636  000010                          .WORD   DPE             ;MASK FOR RMER2
8615  037640  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8616  037642  000240                          NOP                     ;RETURN HERE IF ERROR
8617  037644  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8618  037646  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8619  037650  000137  037670                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8620  037654                          150$:
8621  037654  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8622  037660  000403                          BR      160$                    ;GO TO 160$ IF NO ERROR
8623  037662  000240                          NOP                     ;RETURN HERE IF ERROR
8624  037664  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8625  037666  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8626  037670  062737  000400  001404  160$:   ADD     #400,RMDAO      ;INCREMENT TRACK ADDRESS
8627  037676  022737  004000  001404          CMP     #4000,RMDAO     ;DONE ALL TRACKS??
8628  037704  101012                          BHI     170$            ;NO!!
8629  037706  032737  010000  001430          BIT     #FMT16,RMOFO    ;DONE 16 BIT MODE??
8630  037714  001015                          BNE     180$            ;YES!!
8631  037716  012737  010000  001430          MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
8632  037724  012737  002400  001404          MOV     #002400,RMDAO   ;START WITH TRACK 5
8633  037732                          170$:                   ;TEST NEXT TRACK
8634  037732  004737  053360                  JSR     PC,CNTCLR
8635  037736  000402                          BR      175$    ;GO TO 175$ IF NO ERROR
8636  037740  000240                          NOP                     ;RETURN HERE IF ERROR
8637  037742  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
8638  037744  000137  037514          175$:   JMP     90$
8639
8640  037750                          180$:
8641
8642
8643
8644                                    ;;*****************************************************************
8645                                    ;*TEST 66       SEARCH INVALID CYLINDER
8646                                    ;;*****************************************************************
8647  037750                          TST66:
8648  037750  000004                          SCOPE                   ;SCOPE CALL
8649  037752  000240                          NOP                     ;START OF TEST
8650  037754  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8651  037760  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8652  037764  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8653  037770  012737  000066  001226          MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8654
8655  037776  004737  043216                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8656  040002  054130                          .WORD   054130  ;TASK DESCRIPTOR
8657  040004  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8658  040006  000240                          NOP                     ;RETURN HERE IF ERROR
8659  040010  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8660  040012  000137  040354                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8661  040016  012737  000000  001430  80$:    MOV     #0,RMOFO        ;SET 18 BIT FORMAT
8662  040024  012737  000000  001404          MOV     #0,RMDAO        ;TRACK=SECTOR=0
```

```
8663  040032  012737  001467  001432          MOV     #823.,RMDC0     ;START AT CYLINDER  823
8664  040040  012737  000031  001376          MOV     #SEARCH!GO,RMCS10;SEARCH COMMAND
8665  040046  012702  001533                  MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
8666  040052  112722  000032                  MOVB    #RMOF,(R2)+
8667  040056  112722  000034                  MOVB    #RMDC,(R2)+
8668  040062  112722  000006                  MOVB    #RMDA,(R2)+
8669  040066  112722  000000                  MOVB    #RMCS1,(R2)+
8670  040072  112722  000200                  MOVB    #200,(R2)+
8671  040076  004737  044132                  JSR     PC,GETSTS       ;SETUP STATUS FETCH
8672  040102                          90$:
8673  040102  004737  044466                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8674  040106  000404                          BR      100$    ;GO TO 100$ IF NO ERROR
8675  040110  000240                          NOP                     ;RETURN HERE IF ERROR
8676  040112  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
8677  040114  000137  040276                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8678  040120  004737  045026          100$:   JSR     PC,TIMOUT
8679  040124  004737  044216                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8680  040130  000404                          BR      110$    ;GO TO 110$ IF NO ERROR
8681  040132  000240                          NOP                     ;RETURN HERE IF ERROR
8682  040134  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
8683  040136  000137  040276                  JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8684  040142                          110$:
8685  040142  004737  045212                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8686  040146  000405                          BR      120$            ;GO TO 120$ IF NO ERROR
8687  040150  000240                          NOP                     ;RETURN HERE IF ERROR
8688  040152  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8689  040154  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8690  040156  000137  040276                  JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8691  040162  032737  002000  001342  120$:   BIT     #IAE,RMER1I     ;IS "IAE" SET??
8692  040170  001012                          BNE     130$            ;YES!!
8693  040172  012737  002000  001140          MOV     #IAE,$GDDAT     ;EXPECTED STATUS FOR TYPEOUT
8694  040200  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS FOR TYPEOUT
8695  040206  042737  175777  001142          BIC     #^CIAE,$BDDAT
8696  040214  104257                          ERROR   257             ;IAE NOT SET DURING SEARCH
8697  040216                          130$:
8698  040216  004737  061062                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
8699  040222  000405                          BR      140$            ;GO TO 140$ IF NO ERROR
8700  040224  000240                          NOP                     ;RETURN HERE IF ERROR
8701  040226  104000                          ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
8702  040230  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8703  040232  000137  040276                  JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8704  040236                          140$:
8705  040236  004737  061566                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8706  040242  117760                          .WORD   IAE!NDTMSK      ;MASK FOR RMER1
8707  040244  000010                          .WORD   DPE             ;MASK FOR RMER2
8708  040246  000405                          BR      150$            ;GO TO 150$ IF NO ERROR
8709  040250  000240                          NOP                     ;RETURN HERE IF ERROR
8710  040252  104000                          ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8711  040254  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8712  040256  000137  040276                  JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8713  040262                          150$:
8714  040262  004737  046044                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8715  040266  000403                          BR      160$            ;GO TO 160$ IF NO ERROR
8716  040270  000240                          NOP                     ;RETURN HERE IF ERROR
8717  040272  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8718  040274  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
8719  040276  005237  001432          160$:   INC     RMDCO                   ;INCREMENT CYLINDER ADDRESS
8720  040302  022737  002000  001432          CMP     #1024.,RMDCO            ;DONE ALL CYLINDERS??
8721  040310  101012                          BHI     170$                    ;NO!!
8722  040312  032737  010000  001430          BIT     #FMT16,RMOFO            ;SET 16 BIT FORMAT
8723  040320  001015                          BNE     190$                    ;YES!!
8724  040322  012737  010000  001430          MOV     #FMT16,RMOFO            ;SET 16 BIT FORMAT
8725  040330  012737  001467  001432          MOV     #823.,RMDCO             ;CYLINDER=823
8726  040336                          170$:
8727  040336  004737  053360                  JSR     PC,CNTCLR
8728  040342  000402                          BR      180$    ;GO TO 180$ IF NO ERROR
8729  040344  000240                          NOP                             ;RETURN HERE IF ERROR
8730  040346  104000                          ERROR                           ;ERROR NUMBER DEFINED BY SUB
8731  040350  000137  040102          180$:   JMP     90$
8732  040354                          190$:
8733
8734
8735
8736
8737                                    ;;*****************************************************************
8738                                    ;*TEST 67        IVC SEARCH TEST
8739                                    ;*****************************************************************
8740
8741  040354                          TST67:
8742  040354  000004                          SCOPE                           ;SCOPE CALL
8743  040356  000240                          NOP                             ;START OF TEST
8744  040360  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
8745  040364  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
8746  040370  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
8747  040374  012737  000067  001226          MOV     #67,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
8748
8749  040402  004737  043216                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8750  040406  054130                          .WORD   054130  ;TASK DESCRIPTOR
8751  040410  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
8752  040412  000240                          NOP                             ;RETURN HERE IF ERROR
8753  040414  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8754  040416  000137  040734                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8755  040422                          80$:
8756  040422  112737  000024  001533          MOVB    #RMMR1,PUTINX           ;SETUP PUT INDEX TABLE
8757  040430  112737  000200  001534          MOVB    #200,PUTINX+1           ;WRITE TERMINATOR BYTE
8758  040436  012737  000001  001422          MOV     #DMD,RMMR1O             ;RMMR1=DMD
8759  040444  004737  044466                  JSR     PC,PUT                  ;GO WRITE RMMR1 VIA SUB
8760  040450  000404                          BR      90$                     ;GO TO 90$ IF NO ERROR
8761  040452  000240                          NOP                             ;RETURN HERE IF ERROR
8762  040454  104000                          ERROR                           ;ERROR NUMBER DEFINED BY PUT SUB
8763  040456  000137  040734                  JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8764  040462  012737  000000  001422  90$:     MOV     #0,RMMR1O               ;RESET DIAGNOSTIC MODE
8765  040470  012737  000000  001432          MOV     #0,RMDCO                ;CYLINDER 0
8766  040476  012737  000000  001404          MOV     #0,RMDAO                ;TRACK=SECTOR=0
8767  040504  012737  000000  001430          MOV     #0,RMOFO                ;18 BIT FORMAT
8768  040512  012737  000031  001376          MOV     #SEARCH!GO,RMCS1O;SEARCH COMMAND
8769  040520  012702  001533                  MOV     #PUTINX,R2              ;LOAD INDEX TABLE
8770  040524  112722  000024                  MOVB    #RMMR1,(R2)+
8771  040530  112722  000034                  MOVB    #RMDC,(R2)+
8772  040534  112722  000006                  MOVB    #RMDA,(R2)+
8773  040540  112722  000032                  MOVB    #RMOF,(R2)+
8774  040544  112722  000000                  MOVB    #RMCS1,(R2)+
```

```
8775  040550  112722  000200                      MOVB    #200,(R2)+
8776  040554  004737  044132                      JSR     PC,GETSTS
8777  040560  004737  044466                      JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8778  040564  000404                              BR      100$    ;GO TO 100$ IF NO ERROR
8779  040566  000240                              NOP                     ;RETURN HERE IF ERROR
8780  040570  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
8781  040572  000137  040734                      JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8782  040576  004737  045026          100$:       JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8783  040602  004737  044216                      JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8784  040606  000404                              BR      110$    ;GO TO 110$ IF NO ERROR
8785  040610  000240                              NOP                     ;RETURN HERE IF ERROR
8786  040612  104000                              ERROR                   ;ERROR DEFINED BY GET SUB
8787  040614  000137  040734                      JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8788  040620                          110$:
8789  040620  004737  045212                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8790  040624  000405                              BR      120$            ;GO TO 120$ IF NO ERROR
8791  040626  000240                              NOP                     ;RETURN HERE IF ERROR
8792  040630  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8793  040632  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8794  040634  000137  040734                      JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8795  040640  032737  010000  001370  120$:       BIT     #IVC,RMER2I     ;DID IVC SET??
8796  040646  001012                              BNE     130$            ;YES!!
8797  040650  012737  010000  001140              MOV     #IVC,$GDDAT     ;EXPECTED STATUS
8798  040656  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
8799  040664  042737  167777  001142              BIC     #^CIVC,$BDDAT
8800  040672  104260                              ERROR   260             ;IVC NOT SET DURING SEARCH
8801  040674                          130$:
8802  040674                          140$:
8803  040674  004737  061566                      JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8804  040700  115760                              .WORD   NDTMSK          ;MASK FOR RMER1
8805  040702  010010                              .WORD   IVC!DPE         ;MASK FOR RMER2
8806  040704  000405                              BR      150$            ;GO TO 150$ IF NO ERROR
8807  040706  000240                              NOP                     ;RETURN HERE IF ERROR
8808  040710  104000                              ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8809  040712  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8810  040714  000137  040734                      JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8811  040720                          150$:
8812  040720  004737  046044                      JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8813  040724  000403                              BR      160$            ;GO TO 160$ IF NO ERROR
8814  040726  000240                              NOP                     ;RETURN HERE IF ERROR
8815  040730  104000                              ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8816  040732  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8817  040734                          160$:
8818                                  ;;*****************************************************************
8819                                  ;*TEST 70        ABORT SEARCH TEST
8820                                  ;;*****************************************************************
8821  040734                          TST70:
8822  040734  000004                              SCOPE                   ;SCOPE CALL
8823  040736  000240                              NOP                     ;START OF TEST
8824  040740  012706  001100                      MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8825  040744  013700  001276                      MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8826  040750  013701  001446                      MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8827  040754  012737  000070  001226              MOV     #70,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8828
8829  040762  004737  043216                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8830  040766  054130                              .WORD   054130  ;TASK DESCRIPTOR
```

```
8831  040770  000404                        BR      80$             ;GO TO 80$ IF NO ERROR
8832  040772  000240                        NOP                     ;RETURN HERE IF ERROR
8833  040774  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8834  040776  000137  041300                JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8835  041002  012737  040000  001412  80$:  MOV     #UNS,RMER1O     ;SET UNSAFE ERROR
8836  041010  012737  000031  001376        MOV     #SEARCH!GO,RMCS1O;SEARCH COMMAND
8837  041016  012737  000000  001432        MOV     #0,RMDCO        ;CYLINDER 0
8838  041024  012737  000000  001404        MOV     #0,RMDAO        ;SECTOR=TRACK=0
8839  041032  012702  001533                MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
8840  041036  112722  000034                MOVB    #RMDC,(R2)+
8841  041042  112722  000006                MOVB    #RMDA,(R2)+
8842  041046  112722  000014                MOVB    #RMER1,(R2)+
8843  041052  112722  000000                MOVB    #RMCS1,(R2)+
8844  041056  112722  000200                MOVB    #200,(R2)+      ;TERMINATE TABLE
8845  041062  004737  044132                JSR     PC,GETSTS       ;SETUP FOR STATUS
8846  041066  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
8847  041072  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
8848  041074  000240                        NOP                     ;RETURN HERE IF ERROR
8849  041076  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
8850  041100  000137  041300                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8851  041104                          90$:
8852  041104  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8853  041110  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
8854  041112  000240                        NOP                     ;RETURN HERE IF ERROR
8855  041114  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
8856  041116  000137  041300                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8857  041122  032737  020000  001340  100$: BIT     #PIP,RMDSI      ;IS PIP SET??
8858  041130  001412                        BEQ     110$            ;NO!!
8859  041132  012737  000000  001140        MOV     #0,$GDDAT       ;EXPECTED STATUS
8860  041140  013737  001340  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
8861  041146  042737  157777  001142        BIC     #^CPIP,$BDDAT
8862  041154  104261                        ERROR   261             ;DRIVE DID NOT ABORT SEARCH
8863  041156  004737  045026          110$: JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
8864  041162  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8865  041166  000404                        BR      120$    ;GO TO 120$ IF NO ERROR
8866  041170  000240                        NOP                     ;RETURN HERE IF ERROR
8867  041172  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
8868  041174  000137  041300                JMP     160$    ;GO TO 160$ IF ERROR WAS FOUND
8869  041200                          120$:
8870  041200  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8871  041204  000405                        BR      130$            ;GO TO 130$ IF NO ERROR
8872  041206  000240                        NOP                     ;RETURN HERE IF ERROR
8873  041210  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8874  041212  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8875  041214  000137  041300                JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8876  041220                          130$:
8877  041220  004737  061062                JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
8878  041224  000405                        BR      140$            ;GO TO 140$ IF NO ERROR
8879  041226  000240                        NOP                     ;RETURN HERE IF ERROR
8880  041230  104000                        ERROR                   ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
8881  041232  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8882  041234  000137  041300                JMP     160$            ;GO TO 160$ IF ERROR WAS FOUND
8883  041240                          140$:
8884  041240  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8885  041244  155760                        .WORD   ND!MSK!UNS              ;MASK FOR RMER1
8886  041246  000010                        .WORD   DPE             ;MASK FOR RMER2
```

```
8887  041250  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
8888  041252  000240                        NOP                             ;RETURN HERE IF ERROR
8889  041254  104000                        ERROR                           ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8890  041256  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8891  041260  000137  041300                JMP     160$                    ;GO TO 160$ IF ERROR WAS FOUND
8892  041264                        150$:
8893  041264  004737  046044                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
8894  041270  000403                        BR      160$                    ;GO TO 160$ IF NO ERROR
8895  041272  000240                        NOP                             ;RETURN HERE IF ERROR
8896  041274  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
8897  041276  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8898  041300                        160$:
8899
8900                                  ;;*****************************************************************
8901                                  ;*TEST 71        SEARCH AT OFFSET
8902                                  ;;*****************************************************************
8903  041300                          TST71:
8904  041300  000004                        SCOPE                           ;SCOPE CALL
8905  041302  000240                        NOP                             ;START OF TEST
8906  041304  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
8907  041310  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
8908  041314  013701  001446                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
8909  041320  012737  000071  001226        MOV     #71,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
8910  041326  004737  043216                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8911  041332  054130                        .WORD   054130  ;TASK DESCRIPTOR
8912  041334  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
8913  041336  000240                        NOP                             ;RETURN HERE IF ERROR
8914  041340  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8915  041342  000137  041642                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8916  041346  012737  000000  001404  80$:   MOV     #0,RMDA0                ;TRACK=SECTOR=0
8917  041354  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER=0
8918  041362  012737  010000  001430        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8919  041370  012737  000015  001376        MOV     #OFFSET!GO,RMCS10       ;OFFSET COMMAND
8920  041376  012702  001533                MOV     #PUTINX,R2              ;SETUP REGISTER INDEX TABLE
8921  041402  112722  000006                MOVB    #RMDA,(R2)+             ;FOR SEEK TO CYLINDER 0
8922  041406  112722  000034                MOVB    #RMDC,(R2)+
8923  041412  112722  000032                MOVB    #RMOF,(R2)+
8924  041416  112722  000000                MOVB    #RMCS1,(R2)+
8925  041422  112722  000200                MOVB    #200,(R2)+
8926  041426  004737  044466                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
8927  041432  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
8928  041434  000240                        NOP                             ;RETURN HERE IF ERROR
8929  041436  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
8930  041440  000137  041642                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8931  041444  004737  044132          90$:  JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
8932  041450  004737  045026                JSR     PC,TIMOUT
8933  041454  112737  000000  001533  120$:  MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
8934  041462  112737  000200  001534        MOVB    #200,PUTINX+1
8935  041470  012737  000031  001376        MOV     #SEARCH!GO,RMCS10;STORE SEARCH COMMAND
8936  041476  004737  044466                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
8937  041502  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
8938  041504  000240                        NOP                             ;RETURN HERE IF ERROR
8939  041506  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
8940  041510  000137  041642                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8941  041514                        130$:
8942  041514  004737  045026                JSR     PC,TIMOUT               ;WAIT FOR SEARCH TO COMPLETE
```

```
8943  041520  004737  044216          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
8944  041524  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
8945  041526  000240                  NOP                     ;RETURN HERE IF ERROR
8946  041530  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
8947  041532  000137  041642          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
8948  041536                  140$:
8949  041536  004737  045212          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8950  041542  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
8951  041544  000240                  NOP                     ;RETURN HERE IF ERROR
8952  041546  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8953  041550  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8954  041552  000137  041642          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8955  041556                  150$:
8956  041556  004737  057516          JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
8957  041562  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
8958  041564  000240                  NOP                     ;RETURN HERE IF ERROR
8959  041566  104000                  ERROR                   ;ERROR # DEFINED BY SCHSTS SUBROUTINE
8960  041570  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8961  041572  000137  041642          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8962  041576                  160$:
8963  041576  004737  061566          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
8964  041602  115760                  .WORD   ND!MSK          ;MASK FOR RMER1
8965  041604  000010                  .WORD   DPE             ;MASK FOR RMER2
8966  041606  000405                  BR      170$                    ;GO TO 170$ IF NO ERROR
8967  041610  000240                  NOP                     ;RETURN HERE IF ERROR
8968  041612  104000                  ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
8969  041614  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8970  041616  000137  041642          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8971  041622                  170$:
8972  041622  004737  046044          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8973  041626  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
8974  041630  000240                  NOP                     ;RETURN HERE IF ERROR
8975  041632  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8976  041634  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8977  041636  000137  041642          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
8978  041642                  180$:
8979  041642                  190$:
8980
8981  041642  000137  042150          JMP     $EOSP           ;SKIP OVER HEAD ALIGNMENT SEEK
8982
8983                          ;PUT NEWTEST HERE
8984                          ;;*****************************************************************
8985                          ;*TEST 72        HEAD ALIGNMENT SEEK
8986                          ;;*****************************************************************
8987  041646                  ST72:
8988  041646  000004                  SCOPE                   ;SCOPE CALL
8989  041650  000240                  NOP                     ;START OF TEST
8990  041652  012737  000001  001206          MOV     #1,$TIMES       ;LOAD ITERATION COUNT
8991  041660  012737  041674  001122          MOV     #1$,$LPADR
8992  041666  012737  041674  001124          MOV     #1$,$LPERR
8993  041674                  1$:
8994  041674  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8995  041700  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8996  041704  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8997  041710  012737  000072  001226          MOV     #72,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8998
```

```
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;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.
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

9010  041716                        10$:
9011  041716  004737  043216                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
9012  041722  054130                        .WORD   054130  ;TASK DESCRIPTOR
9013  041724  000404                        BR      80$             ;GO TO 80$ IF NO ERROR
9014  041726  000240                        NOP                     ;RETURN HERE IF ERROR
9015  041730  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
9016  041732  000137  042150                JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
9017  041736                        80$:
9018  041736  012737  000000  001404        MOV     #0,RMDAO        ;TRACK = SECTOR = 0
9019  041744  012737  000365  001432        MOV     #245.,RMDCO     ;CYLINDER = 245
9020  041752  012737  000005  001376        MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
9021  041760  012702  001533                MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
9022  041764  112722  000034                MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
9023  041770  112722  000006                MOVB    #RMDA,(R2)+
9024  041774  112722  000000                MOVB    #RMCS1,(R2)+
9025  042000  112722  000200                MOVB    #200,(R2)+      ;WRITE TERMINATOR
9026  042004  004737  044466                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
9027  042010  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
9028  042012  000240                        NOP                     ;RETURN HERE IF ERROR
9029  042014  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
9030  042016  000137  042150                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
9031  042022  004737  044132        90$:    JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
9032  042026  004737  045026                JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
9033  042032  004737  044216                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
9034  042036  000404                        BR      100$    ;GO TO 100$ IF NO ERROR
9035  042040  000240                        NOP                     ;RETURN HERE IF ERROR
9036  042042  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
9037  042044  000137  042150                JMP     140$    ;GO TO 140$ IF ERROR WAS FOUND
9038  042050                        100$:
9039  042050  004737  045212                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
9040  042054  000405                        BR      110$            ;GO TO 110$ IF NO ERROR
9041  042056  000240                        NOP                     ;RETURN HERE IF ERROR
9042  042060  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
9043  042062  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9044  042064  000137  042150                JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
9045  042070                        110$:
9046  042070  004737  052120                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
9047  042074  000405                        BR      120$            ;GO TO 120$ IF NO ERROR
9048  042076  000240                        NOP                     ;RETURN HERE IF ERROR
9049  042100  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
9050  042102  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9051  042104  000137  042150                JMP     140$            ;GO TO 140$ IF ERROR WAS FOUND
9052  042110                        120$:
9053  042110  004737  061566                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
9054  042114  115760                        .WORD   ND!MSK          ;MASK FOR RMER1
```

```
 9055  042116  000010                      .WORD   DPE             ;MASK FOR RMER2
 9056  042120  000405                      BR      130$                    ;GO TO 130$ IF NO ERROR
 9057  042122  000240                      NOP                     ;RETURN HERE IF ERROR
 9058  042124  104000                      ERROR                   ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 9059  042126  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 9060  042130  000137  042150              JMP     140$                    ;GO TO 140$ IF ERROR WAS FOUND
 9061  042134                      130$:
 9062  042134  004737  046044              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 9063  042140  000403                      BR      140$                    ;GO TO 140$ IF NO ERROR
 9064  042142  000240                      NOP                     ;RETURN HERE IF ERROR
 9065  042144  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 9066  042146  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 9067  042150                      140$:
```

```
                                        $EOSP:
042150                                          NOP
042150  000240                                    MOV     TSTQUE,R0
042152  013700  001446                            ADD     #2,R0
042156  062700  000002                            MOV     R0,TSTQUE
042162  010037  001446                            TST     (R0)
042166  005710                                    BEQ     1$
042170  001402                                    JMP     READY
042172  000137  007106                  1$:       MOV     #TSTQUE+2,TSTQUE
042176  012737  001450  001446        .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

042204                                  $EOP:
042204  000240                                    NOP
042206  005037  001116                            CLR     $TSTNM          ;;ZERO THE TEST NUMBER
042212  005037  001206                            CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
042216  005237  001230                            INC     $PASS           ;;INCREMENT THE PASS NUMBER
042222  042737  100000  001230                    BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
042230  005327                                    DEC     (PC)+           ;;LOOP?
042232  000001                          $EOPCT:   .WORD   1
042234  003063                                    BGT     $DOAGN          ;;YES
042236  012737                                    MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
042240  000001                          $ENDCT:   .WORD   1
042242  042232                                    $EOPCT
042244  104401  042252                            TYPE    ,65$            ;;TYPE ASCIZ STRING
042250  000407                                    BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:    .ASCIZ  <12><15>/END PASS #/
042270                                  64$:
042270  013746  001230                            MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                        ;;TYPE PASS NUMBER
042274  104405                                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
042276  104401  042304                            TYPE    ,67$            ;;TYPE ASCIZ STRING
042302  000421                                    BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:    .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
042346                                  66$:
042346  013746  001126                            MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                        ;;TOTAL NUMBER OF ERRORS
042352  104405                                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
042354  104401  001217                            TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
042360  005037  001126                            CLR     $ERTTL          ;;CLEAR ERROR TOTAL
042364  013700  000042                  $GET42:   MOV     @#42,R0         ;;GET MONITOR ADDRESS
042370  001405                                    BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
042372  000005                                    RESET                   ;;CLEAR THE WORLD
042374  004710                          $ENDAD:   JSR     PC,(R0)         ;;GO TO MONITOR
042376  000240                                    NOP                     ;;SAVE ROOM
042400  000240                                    NOP                     ;;FOR
042402  000240                                    NOP                     ;;ACT11
042404                                  $DOAGN:
042404  000137                                    JMP     @(PC)+          ;;RETURN
042406  007106                          $RTNAD:   .WORD   READY
```

```
 9124  042410      377      377      000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 9125                042414                       .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.
042414                                  ERRTYP:
042414  104414                                  SAVREG
042416  032777  020000  136530                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
042424  001402                                  BEQ     1$              ;NO!!
042426  000137  043144                          JMP     21$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
042432  104401  001217                1$:       TYPE    ,$CRLF
042436  104401  043160                          TYPE    ,ERTY00         ;TYPE "UNT#"
042442  013746  001234                          MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
042446  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042450     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042451     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042452  005037  043150                          CLR     TSTNMB          ;LOAD TEST NUMBER FOR
042456  013737  001226  043150                  MOV     $TESTN,TSTNMB
042464  104401  043166                          TYPE    ,ERTY01         ;TYPE "TST#"
042470  013746  043150                          MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
042474  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042476     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042477     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042500  005037  043152                          CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
042504  113737  001130  043152                  MOVB    $ITEMB,ERRNMB   ;TYPEOUT
042512  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
042514  104401  043176                          TYPE    ,ERTY02         ;TYPE "ERR#"
042520  013746  043152                          MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
042524  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042526     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
042527     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
042530  104401  043205                2$:       TYPE    ,ERTY03         ;TYPE "PC="
042534  013746  001132                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
042540  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
042542     006                                  .BYTE   6               ;;TYPE 6 DIGIT(S)
042543     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
042544  005737  043152                3$:       TST     ERRNMB          ;WAS AN ERROR CALLED?
042550  001575                                  BEQ     21$             ;NO!!
042552  104401  001217                          TYPE    ,$CRLF          ;YES-TYPE CRLF
042556  105037  043156                          CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
9182  042562  105037  043157                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
9183  042566  013700  043152                  MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
9184  042572  006300                          ASL     R0              ;FOUR ENTRIES IN ERROR
9185  042574  006300                          ASL     R0              ;TABLE
9186  042576  006300                          ASL     R0
9187  042600  062700  001552                  ADD     #$ERRTB-8.,R0
9188  042604  011001                          MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
9189                                                                  ;TABLE
9190  042606  001507                          BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
9191                                ;TYPE THE ERROR MESSAGE
9192  042610  012102                  4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
9193  042612  001505                          BEQ     12$             ;BRANCH IF END OF MESSAGE
9194  042614  010237  042762                  MOV     R2,11$          ;LOAD ADDRESS OF STRING
9195  042620  005037  043154                  CLR     BOTADR          ;CLEAR BOT ADDRESS
9196  042624  112203                  5$:     MOVB    (R2)+,R3        ;END OF STRING??
9197  042626  001454                          BEQ     10$             ;YES!!
9198  042630  122703  000015                  CMPB    #CR,R3          ;CARRIAGE RETURN??
9199  042634  001003                          BNE     6$              ;NO!!
9200  042636  105037  043157                  CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
9201  042642  000770                          BR      5$              ;GET NEXT CHARACTER
9202  042644  122703  000012          6$:     CMPB    #LF,R3          ;LINE FEED??
9203  042650  001765                          BEQ     5$              ;YES-GET NEXT CHARACTER
9204  042652  122703  000011                  CMPB    #HT,R3          ;HORIZONTAL TAB??
9205  042656  001007                          BNE     8$              ;NO!!
9206  042660  105237  043157          7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
9207  042664  132737  000007  043157          BITB    #7,CHRCNT
9208  042672  001372                          BNE     7$
9209  042674  000407                          BR      9$
9210  042676  105237  043157          8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
9211  042702  122703  000040                  CMPB    #' ,R3          ;SPACE??
9212  042706  001002                          BNE     9$              ;NO!!
9213  042710  010237  043154                  MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
9214  042714  122737  000100  043157  9$:     CMPB    #64.,CHRCNT     ;END OF LINE??
9215  042722  103340                          BHIS    5$              ;NO!!
9216  042724  013704  043154                  MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
9217  042730  001007                          BNE     90$             ;BRANCH IF SPACE DETECTED
9218  042732  104401  001217                  TYPE    ,$CRLF          ;TYPE CRLF
9219  042736  105037  043157                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
9220  042742  013702  042762                  MOV     11$,R2          ;SET UP R2 FOR TESTING
9221  042746  000726                          BR      5$
9222  042750  105044                  90$:    CLRB    -(R4)           ;REPLACE SPACE
9223  042752  112737  177777  043156          MOVB    #-1,BOTFLG      ;SET BOT FLAG
9224  042760  104401                  10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
9225  042762  000000                  11$:    .WORD                   ;STRING ADDRESS GOES HERE
9226  042764  105737  043156                  TSTB    BOTFLG          ;WAS STRING TRUNCATED??
9227  042770  001707                          BEQ     4$              ;NO!!
9228  042772  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
9229  042776  105037  043156                  CLRB    BOTFLG          ;CLEAR BOT FLAG
9230  043002  105037  043157                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
9231  043006  013702  043154                  MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
9232  043012  010237  042762                  MOV     R2,11$          ;SETUP 11$ FOR TYPING
9233  043016  112742  000040                  MOVB    #' ,-(R2)       ;RESTORE SPACE
9234  043022  105722                          TSTB    (R2)+           ;RESTORE R2
9235  043024  000677                          BR      5$              ;TYPE REST OF STRING
9236  043026                          12$:
9237                                ;TYPE ERROR HEADER AND ERROR DATA
```

```
9238  043026                        13$:
9239  043026  016001  000002              MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
9240  043032  001444                      BEQ     21$             ;BRANCH IF NO HEADER
9241  043034  104401  001217              TYPE    ,$CRLF          ;(ASSUME NO DATA)
9242  043040  016002  000004              MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
9243  043044  016003  000006              MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
9244  043050  012137  043060      14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
9245  043054  001433                      BEQ     21$             ;BRANCH IF END OF HEADERS
9246                                                              ;(ASSUME END OF DATA)
9247  043056  104401                      TYPE
9248  043060  000000              15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
9249  043062  104401  001217              TYPE    ,$CRLF
9250  043066  005702                      TST     R2              ;DATA WITH HEADER??
9251  043070  001767                      BEQ     14$             ;NO!!
9252  043072  012204                      MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
9253  043074  012305                      MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
9254  043076  105725              16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
9255  043100  100407                      BMI     18$             ;BINARY
9256  043102  001403                      BEQ     17$             ;OCTAL
9257  043104  013446                      MOV     @(R4)+,-(SP)    ;DECIMAL
9258  043106  104405                      TYPDS
9259  043110  000405                      BR      19$
9260  043112  013446              17$:    MOV     @(R4)+,-(SP)
9261  043114  104402                      TYPOC
9262  043116  000402                      BR      19$
9263  043120  013446              18$:    MOV     @(R4)+,-(SP)
9264  043122  104406                      TYPBN
9265  043124  005714              19$:    TST     (R4)            ;MORE DATA??
9266  043126  001403                      BEQ     20$             ;NO!!
9267  043130  104401  043213              TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
9268  043134  000760                      BR      16$             ;AND CONTINUE
9269  043136  104401  001217      20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
9270  043142  000742                      BR      14$             ;BEFORE NEXT HEADER
9271  043144  104415              21$:    RESREG
9272  043146  000207                      RTS     PC
9273
9274  043150  000000              TSTNMB: .WORD   0               ;TEST NUMBER
9275  043152  000000              ERRNMB: .WORD   0               ;ERROR NUMBER
9276  043154  000000              BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
9277  043156     000              BOTFLG: .BYTE   0               ;BOT FLAG
9278  043157     000              CHRCNT: .BYTE   0               ;CHARACTER COUNT
9279
9280  043160  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
9281  043166  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
9282  043174  000043
9283  043176  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
9284  043204     000
9285  043205     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
9286  043212     000
9287  043213     040  000040          ERTY04: .ASCIZ  @ @
9288                                  .EVEN
9289
```

```
                                .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

043216                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
043216  017637  000000  044130          MOV     @(SP),500$      ;STORE DESCRIPTOR
043224  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
043230  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
043234  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
043240  032737  100000  044130          BIT     #BIT15,500$     ;SELECT DEVICE??
043246  001414                          BEQ     30$             ;NO!!
043250  004737  051706                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
043254  000411                          BR      30$             ;NO ERROR - CONTINUE
043256  000401                          BR      20$
043260  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
043262  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
043266  113776  043260  000000          MOVB    10$,@(SP)
043274  000137  044116                  JMP     400$
043300                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
043300  032737  040000  044130          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
043306  001453                          BEQ     120$            ;NO!!
```

```
 9346  043310  004737  053360                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 9347  043314  000411                          BR      60$             ;CONTINUE - NO ERROR
 9348  043316  000401                          BR      50$
 9349  043320  000000                  40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
 9350  043322  062716  000004          50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9351  043326  113776  043320  000000          MOVB    40$,@(SP)
 9352  043334  000137  044116                  JMP     400$
 9353  043340                          60$:
 9354                                  ;;*****************************************************************
 9355                                  ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
 9356  043340  032737  000100  044130          BIT     #BIT6,500$      ;VERIFY??
 9357  043346  001433                          BEQ     120$            ;NO!!
 9358  043350  004737  044132                  JSR     PC,GETSTS       ;SETUP INDEX TABLE
 9359  043354  004737  044216                  JSR     PC,GET          ;GO GET STATUS
 9360  043360  000411                          BR      90$             ;NO ERROR GETTING STATUS
 9361  043362  000401                          BR      80$
 9362  043364  000000                  70$:    .WORD                   ;ERROR FROM GETTING STATUS
 9363  043366  062716  000004          80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9364  043372  113776  043364  000000          MOVB    70$,@(SP)
 9365  043400  000137  044116                  JMP     400$
 9366  043404  004737  053476          90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
 9367  043410  000412                          BR      120$            ;NO ERROR IN CLEAR
 9368  043412  000401                          BR      110$
 9369  043414  000000                  100$:   .WORD                   ;ERROR IN STATUS CLEAR
 9370  043416  005726                  110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
 9371  043420  062716  000004                  ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
 9372  043424  113776  043414  000000          MOVB    100$,@(SP)      ;ERROR TO USER
 9373  043432  000137  044116                  JMP     400$
 9374  043436                          120$:
 9375
 9376                                  ;;*****************************************************************
 9377                                  ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
 9378                                  ;NOT VALID
 9379  043436  032737  010000  044130          BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
 9380  043444  001511                          BEQ     240$            ;NO!!
 9381  043446  112737  000012  001504          MOVB    #RMDS,GETINX    ;GET RMDS
 9382  043454  112737  000200  001505          MOVB    #200,GETINX+1
 9383  043462  004737  044216                  JSR     PC,GET
 9384  043466  000411                          BR      150$            ;NO ERROR GETTING RMDS
 9385  043470  000401                          BR      140$
 9386  043472  000000                  130$:   .WORD
 9387  043474  062716  000004          140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9388  043500  113776  043472  000000          MOVB    130$,@(SP)
 9389  043506  000137  044116                  JMP     400$
 9390  043512  032737  000100  001340  150$:   BIT     #VV,RMDSI       ;IS VOLUME VALID??
 9391  043520  001063                          BNE     240$            ;YES!!
 9392  043522  005037  001472                  CLR     MEDENB          ;CLEAR MEDIA ENABLE
 9393  043526  012737  000023  001376          MOV     #PAKACK!GO,RMCS1@       ;LOAD PACK ACK COMMAND
 9394  043534  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
 9395  043542  112737  000200  001534          MOVB    #200,PUTINX+1
 9396  043550  004737  044466                  JSR     PC,PUT          ;GO WRITE COMMAND
 9397  043554  000410                          BR      180$            ;NO ERROR LOADING REGISTER
 9398  043556  000401                          BR      170$
 9399  043560  000000                  160$:   .WORD                   ;ERROR FROM PUT SUB
 9400  043562  062716  000004          170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9401  043566  113776  043560  000000          MOVB    160$,@(SP)
```

```
 9402  043574  000550                          BR      400$
 9403  043576                          180$:
 9404
 9405                                  ;;*****************************************************************
 9406                                  ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
 9407  043576  032737  000020  044130          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
 9408  043604  001431                          BEQ     240$            ;NO!!
 9409  043606  004737  044132                  JSR     PC,GETSTS       ;SETUP FOR STATUS
 9410  043612  004737  044216                  JSR     PC,GET          ;GO GET STATUS
 9411  043616  000410                          BR      210$            ;NO ERROR GETTING STATUS
 9412  043620  000401                          BR      200$
 9413  043622  000000                  190$:   .WORD                   ;ERROR FROM GET SUB
 9414  043624  062716  000004          200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9415  043630  113776  043622  000000          MOVB    190$,@(SP)
 9416  043636  000527                          BR      400$
 9417  043640  004737  054356          210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
 9418  043644  000411                          BR      240$            ;NO ERROR
 9419  043646  000401                          BR      230$
 9420  043650  000000                  220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
 9421  043652  005726                  230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
 9422  043654  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9423  043660  113776  043650  000000          MOVB    220$,@(SP)
 9424  043666  000513                          BR      400$
 9425  043670                          240$:
 9426
 9427                                  ;;*****************************************************************
 9428                                  ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
 9429  043670  032737  004000  044130          BIT     #BIT11,500$     ;RECALIBRATE??
 9430  043676  001513                          BEQ     410$            ;NO!!
 9431  043700  112737  000012  001504          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
 9432  043706  112737  000200  001505          MOVB    #200,GETINX+1
 9433  043714  004737  044216                  JSR     PC,GET          ;GO GET RMDS
 9434  043720  000410                          BR      270$            ;NO ERROR GETTING RMDS
 9435  043722  000401                          BR      260$
 9436  043724  000000                  250$:   .WORD                   ;ERROR FROM GET SUB
 9437  043726  062716  000004          260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9438  043732  113776  043724  000000          MOVB    250$,@(SP)
 9439  043740  000466                          BR      400$
 9440  043742  032737  020000  001340  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
 9441  043750  001466                          BEQ     410$            ;NO!!
 9442  043752  012737  000007  001376          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
 9443  043760  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
 9444  043766  112737  000200  001534          MOVB    #200,PUTINX+1
 9445  043774  004737  044466                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
 9446  044000  000410                          BR      300$            ;NO ERROR
 9447  044002  000401                          BR      290$
 9448  044004  000000                  280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
 9449  044006  062716  000004          290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9450  044012  113776  044004  000000          MOVB    280$,@(SP)
 9451  044020  000436                          BR      400$
 9452  044022  004737  044132          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
 9453  044026  004737  045026                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
 9454
 9455                                  ;;*****************************************************************
 9456                                  ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
 9457  044032  032737  000010  044130          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
9458  044040  001432                    BEQ     410$            ;NO!!
9459  044042  004737  044216            JSR     PC,GET          ;GO GET STATUS
9460  044046  000410                    BR      330$            ;NO ERROR GETTING STATUS
9461  044050  000401                    BR      320$
9462  044052  000000            310$:   .WORD                   ;ERROR FROM GET
9463  044054  062716  000004    320$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9464  044060  113776  044052  000000    MOVB    310$,@(SP)
9465  044066  000413                    BR      400$
9466  044070  004737  055152    330$:   JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
9467  044074  000414                    BR      410$            ;NO ERROR DURING RECALIBRATE
9468  044076  000401                    BR      350$
9469  044100  000000            340$:   .WORD                   ;ERROR DURING RECALIBRATE
9470  044102  005726            350$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
9471  044104  062716  000004            ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9472  044110  113776  044100  000000    MOVB    340$,@(SP)
9473  044116  162716  000002    400$:   SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
9474  044122  000240                    NOP
9475  044124  000240                    NOP
9476  044126  000207            410$:   RTS     PC              ;RETURN TO USER
9477
9478  044130  000000            500$:   .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,GE\STS
                                ;       ???                     RETURN HERE
                                ;
044132                          GETSTS:
044132  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044134  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044136  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044140  012700  001504                  MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
044144  012701  001376                  MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
044150  012702  000046                  MOV     #RMEC2,R2       ;R2 = REGISTER INDE
044154  110220                  2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
044156  005041                          CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
044160  162702  000002          3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
044164  100405                          BMI     4$              ;BRANCH OUT IF DONE
044166  022702  000022                  CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
044172  001370                          BNE     2$
044174  005041                          CLR     -(R1)
044176  000770                          BR      3$
044200  112720  000200          4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
044204  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
044206  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
044210  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
044212  000240                          NOP
044214  000207                          RTS     PC
```

```
                                .SBTTL  GET SUBROUTINE

                                ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
                                ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
                                ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
                                ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
                                ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
                                ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
                                ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
                                ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
                                ;WHICH SHOULD FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;(1)     "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
                                ;       VALUES AND TERMINATED WITH A CONTROL BYTE
                                ;(2)     "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
                                ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
                                ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
                                ;(3)     JSR     PC,GET
                                ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;       ERROR                   SUB DEFINES ERROR NUMBER
                                ;       ???
                                ;
                                GET:    NOP
044216  000240                         ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
044220  062716  000004                  CLRB    @(SP)           ;ERROR CALL
044224  105076  000000                  SUB     #4,(SP)
044230  162716  000004                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044234  010046                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044236  010146                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044240  010246                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044242  010346                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
044244  010446                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044246  013746  000004                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044252  013746  000006                  MOV     $BASE,R0
044256  013700  001276                  MOV     #GETBUF,R2
044262  012702  001326                  MOV     #GETINX,R4
044266  012704  001504                  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
044272  012737  044400  000004          MOV     #PR6,ERRVEC+2
044300  012737  000300  000006          MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
044306  016037  000010  001174  1$:     MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
044314  016037  000000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
044322  032737  004000  001176          BNE     3$              ;YES!!
044330  001007                          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
044332  062766  000004  000016          MOVB    #112,@16(SP)    ;ERROR CALL
044340  112776  000112  000016          BR      7$
044346  000423                  3$:     TSTB    (R4)            ;DONE??
044350  105714                          BMI     9$              ;YES!!
044352  100433                          MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
044354  111401                          BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
044356  042701  177700                  ADD     R0,R1
044362  060001                          MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
044364  112403                          BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
044366  042703  177700                  ADD     R2,R3
044372  060203                          MOV     (R1),(R3)       ;READ REGISTER
044374  011113
```

```
9566  044376  000764                          BR      3$
9567
9568  044400  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
9569  044402  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
9570  044410  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
9571  044416  162766  000002  000016  7$:     SUB     #2,16(SP)
9572  044424  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
9573  044426  100405                          BMI     9$              ;YES!!
9574  044430  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
9575  044432  112403                          MOVB    (R4)+,R3        ;LOCATIONS
9576  044434  060203                          ADD     R2,R3
9577  044436  005013                          CLR     (R3)
9578  044440  000771                          BR      8$
9579  044442                          9$:
9580  044442  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
9581  044446  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
9582  044452  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
9583  044454  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
9584  044456  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
9585  044460  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
9586  044462  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9587  044464  000207                          RTS     PC              ;RETURN
9588
```

```
                                .SBTTL  PUT SUBROUTINE

                                ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
                                ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
                                ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
                                ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
                                ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
                                ;FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;
                                ;(1)     "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
                                ;        OF REGISTERS TO BE WRITTEN.
                                ;(2)     "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
                                ;        REGISTER TO BE WRITTEN.
                                ;(3)     JSR     PC,PUT
                                ;        BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;        NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;        ERROR                   SUB DEFINES ERROR NUMBER
                                ;        ???
                                ;
044466  000240                  PUT:    NOP
044470  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044472  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044474  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044476  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044500  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
044502  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044506  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044512  013700  001276                  MOV     $BASE,R0
044516  012702  001376                  MOV     #PUTBUF,R2
044522  012704  001533                  MOV     #PUTINX,R4
044526  012737  044634  000004          MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
044534  012737  000300  000006          MOV     #PR6,ERRVEC+2
044542  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
044550  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
044556  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
044564  001007                          BNE     3$              ;YES!!
044566  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
044574  112776  000112  000016          MOVB    #112,@16(SP)    ;USER'S ERROR CALL
044602  000423                          BR      7$
044604  105714                  3$:     TSTB    (R4)            ;DONE??
044606  100424                          BMI     9$              ;YES!!
044610  111401                          MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
044612  042701  177700                  BIC     #^C<377>,R1     ;CLEAR ANY SIGN EXTENSION
044616  060001                          ADD     R0,R1
044620  112403                          MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
044622  042703  177700                  BIC     #^C<377>,R3     ;CLEAR ANY SIGN EXTENSION
044626  060203                          ADD     R2,R3
044630  011311                          MOV     (R3),(R1)       ;WRITE REGISTER
044632  000764                          BR      3$

044634  022626                  5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
044636  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
044644  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
044652  162766  000002  000016  7$:     SUB     #2,16(SP)
```

```
9645
9646  044660                          9$:
9647  044660  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
9648  044664  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
9649  044670  012604                          MOV     (SP)+,R4                ;;POP STACK INTO R4
9650  044672  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
9651  044674  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
9652  044676  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
9653  044700  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
9654  044702  000207                          RTS     PC                      ;RETURN
9655
```

```
                                        .SBTTL  SIZE CLOCK SUBROUTINE
                                        
044704                                  SIZCLK:
044704  013746  000004                        MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044710  013746  000006                        MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044714  012737  044752  000004              MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
044722  012737  000300  000006              MOV     #PR6,ERRVEC+2
044730  012737  177546  001500              MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
044736  012737  000100  001502              MOV     #100,CLKVCT
044744  005777  134530                      TST     @CLKADR         ;TEST FOR KW11-L PRESENT
044750  000421                              BR      3$              ;YES - KW11-L IS PRESENT
044752  022626                          1$: CMP     (SP)+,(SP)+     ;RESTORE SP
044754  012737  045004  000004              MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
044762  012737  172540  001500              MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
044770  012737  000104  001502              MOV     #104,CLKVCT
044776  005777  134476                      TST     @CLKADR         ;TEST FOR KW11-P PRESENT
045002  000404                              BR      3$              ;YES - KW11-P IS PRESENT
045004  022626                          2$: CMP     (SP)+,(SP)+     ;RESTORE SP
045006  062766  000002  000004              ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
045014                                  3$:
045014  012637  000006                      MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
045020  012637  000004                      MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
045024  000207                              RTS     PC              ;RETURN TO USER
```

```
                                        .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
                                        ;
045026                                  TIMOUT:
045026  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
045030  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
045032  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
045034  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
045040  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
045044  012737  045146  000004                  MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
045052  012737  000300  000006                  MOV     #PR6,ERRVEC+2
045060  013700  001276                          MOV     $BASE,R0        ;R0=RM03 ADDRESS
045064  013701  001500                          MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
045070  012702  000037                          MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
045074  020127  172540                  1$:     CMP     R1,#172540      ;KW11-P CLOCK??
045100  001003                                  BNE     2$              ;NO!!
045102  012761  000001  000002                  MOV     #1,2(R1)        ;SET COUNTER
045110  012711  000005                  2$:     MOV     #BIT2!BIT0,(R1) ;START COUNTER
                                        ;
045114  016046  000000                  3$:     MOV     RMCS1(R0),-(SP) ;GET STATUS
045120  042716  177576                          BIC     #^C<RDY!GO>,(SP)
045124  022726  000200                          CMP     #RDY,(SP)+      ;RDY=1,GO=0??
045130  001420                                  BEQ     5$              ;YES!!
045132  032711  000200                          BIT     #BIT7,(R1)      ;TIMER DONE??
045136  001766                                  BEQ     3$              ;NO!!
045140  005302                                  DEC     R2              ;DEC NUMBER OF CYCLES
045142  001354                                  BNE     1$              ;CONTINUE IF NOT DONE
045144  000412                                  BR      5$
045146  022626                  4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
045150  062766  000004  000012                  ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
045156  112776  000007  000012                  MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
045164  162766  000002  000012                  SUB     #2,12(SP)
                                        ;
045172                                  5$:
045172  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
045176  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
045202  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
045204  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
045206  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
045210  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., DVA (BIT 11 OF RMCS1) IS SET
                        ;AND NED (BIT 12 OF RMCS2) IS RESET;
                        ;
                        ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
                        ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
                        ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
                        ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
                        ;I.E., MCPE = 0.
                        ;       .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
                        ;I.E., PAR = 0, OR, PAR = DPE = 1
                        ;
                        ;THE SUBROUTINE ASSUMES THAT:
                        ;
                        ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
                        ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
                        ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
                        ;
                        ;       .($UNIT) CONTAINS THE DRIVE NUMBER
                        ;
                        ;THE SUBROUTINE IS CALLED AS FOLLOWS:
                        ;
                        ;(1)     JSR     PC,PRIERR
                        ;       BR      ???             RETURN HERE IF NO ERROR
                        ;       NOP                     RETURN HERE TO REPORT AN ERROR
                        ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                        ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                        ;       ???                     RETURN HERE IF NO MORE ERRORS
                        ;
045212                  PRIERR:
                        ;
                        ;CLEAR USER'S ERROR CALL
045212  062716  000004          ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
045216  105076  000000          CLRB    @(SP)           ;CLEAR ERROR NUMBER
045222  162716  000004          SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                        ;
                        ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
045226  013737  001336  001142  MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
045234  042737  177770  001142  BIC     #^CUNTMSK,$BDDAT
045242  013737  001234  001140  MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
045250  042737  177770  001140  BIC     #^CUNTMSK,$GDDAT
045256  123737  001140  001142  CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
                                                        ;DRIVE NUMBERS
045264  001415                  BEQ     1$              ;YES!!
045266  062716  000004          ADD     #4,(SP)
045272  112776  000001  000000  MOVB    #1,@(SP)        ;ERROR 1
```

```
9781  045300  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9782  045304  004736                  JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
9783  045306  162716  000010          SUB     #10,(SP)         ;RESTORE (SP)
9784  045312  000240                  NOP
9785  045314  000137  046034          JMP     10$              ;SKIP OTHER CHECKS
9786  045320                      1$:
9787
9788                              ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
9789                              ;THE DEVICE IS NONEXISTANT
9790  045320  032737  004000  001326      BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
9791  045326  001045                      BNE     5$               ;YES!!
9792  045330  013737  001326  001140      MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
9793  045336  052737  004000  001140      BIS     #DVA,$GDDAT
9794  045344  013737  001326  001142      MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
9795  045352  062716  000004              ADD     #4,(SP)
9796  045356  112776  000002  000000      MOVB    #2,@(SP)         ;ERROR #2
9797  045364  032737  010000  001336      BIT     #NED,RMCS2I      ;WAS NED SET??
9798  045372  001414                      BEQ     2$               ;NO!!
9799  045374  013737  001336  001140      MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
9800  045402  013737  001336  001142      MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
9801  045410  042737  010000  001140      BIC     #NED,$GDDAT
9802  045416  112776  000003  000000      MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
9803  045424  162716  000002          2$: SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9804  045430  004736                      JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
9805  045432  162716  000010              SUB     #10,(SP)         ;RESTORE (SP)
9806  045436  000240                      NOP
9807  045440  000575                      BR      10$              ;SKIP OTHER CHECKS
9808  045442                      5$:
9809
9810                              ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
9811  045442  032737  000200  001326      BIT     #RDY,RMCS1I      ;CONTROLLER READY??
9812  045450  001030                      BNE     7$               ;YES!!
9813  045452  013737  001326  001140      MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
9814  045460  052737  000200  001140      BIS     #RDY,$GDDAT
9815  045466  042737  160001  001140      BIC     #SC!TRE!MCPE!GO,$GDDAT
9816  045474  013737  001326  001142      MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
9817  045502  062716  000004              ADD     #4,(SP)
9818  045506  112776  000004  000000      MOVB    #4,@(SP)         ;ERROR #4
9819  045514  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9820  045520  004736                      JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
9821  045522  162716  000010              SUB     #10,(SP)         ;RESTORE (SP)
9822  045526  000240                      NOP
9823  045530  000541                      BR      10$              ;SKIP OTHER CHECKS
9824  045532                      7$:
9825
9826                              ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
9827  045532  032737  000001  001326      BIT     #GO,RMCS1I       ;GO RESET??
9828  045540  001431                      BEQ     8$               ;YES!!
9829  045542  032737  000200  001340      BIT     #DRY,RMDSI       ;DRIVE READY??
9830  045550  001025                      BNE     8$               ;YES!!
9831  045552  013737  001326  001140      MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
9832  045560  042737  160001  001140      BIC     #SC!TRE!MCPE!GO,$GDDAT
9833  045566  013737  001326  001142      MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
9834  045574  062716  000004              ADD     #4,(SP)
9835  045600  112776  000005  000000      MOVB    #5,@(SP)         ;ERROR #5
9836  045606  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
 9837  045612  004736                        JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
 9838  045614  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
 9839  045620  000240                        NOP
 9840  045622  000504                        BR      10$
 9841  045624                          8$:
 9842
 9843                                  ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
 9844                                  ;PARITY ON THE MASSBUS CONTROL BUS
 9845  045624  032737  020000  001326        BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
 9846  045632  001425                        BEQ     9$              ;NO!!
 9847  045634  013737  001326  001140        MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
 9848  045642  042737  160001  001140        BIC     #SC!TRE!MCPE!GO,$GDDAT
 9849  045650  013737  001326  001142        MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
 9850  045656  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
 9851  045662  112776  000013  000000        MOVB    #13,@(SP)       ;ERROR #47
 9852  045670  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9853  045674  004736                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9854  045676  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
 9855  045702  000240                        NOP
 9856  045704  000453                        BR      10$
 9857  045706                          9$:
 9858
 9859                                  ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
 9860  045706  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
 9861  045714  001451                        BEQ     11$             ;NO!!
 9862  045716  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
 9863  045724  001045                        BNE     11$             ;NOT SURE!!
 9864  045726  032737  000010  001412        BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
 9865  045734  001413                        BEQ     93$             ;NO!!
 9866  045736  010046                        MOV     R0,-(SP)        ;PUSH R0 ON STACK
 9867  045740  012700  001533                MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
 9868  045744  122710  000014          91$:  CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
 9869  045750  001002                        BNE     92$
 9870  045752  012600                        MOV     (SP)+,R0        ;POP STACK INTO R0
 9871  045754  000431                        BR      11$             ;PAR WAS SET BY TEST
 9872  045756  105720                  92$:  TSTB    (R0)+           ;END OF TABLE??
 9873  045760  100371                        BPL     91$             ;NO!!
 9874  045762  012600                        MOV     (SP)+,R0        ;POP STACK INTO R0
 9875  045764  013737  001342  001140  93$:  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 9876  045772  042737  000010  001140        BIC     #PAR,$GDDAT
 9877  046000  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 9878  046006  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 9879  046012  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
 9880  046020  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9881  046024  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR
 9882  046026  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 9883  046032  000240                        NOP
 9884  046034  062716  000010          10$:  ADD     #10,(SP)                ;RETURN TO ERROR
 9885  046040  000240                  11$:  NOP                     ;RETURN TO NO ERROR
 9886  046042  000207                        RTS     PC
 9887
```

```
                                .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
                                ;
                                ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
                                ;SUCH AS UNEXPECIED 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.
046044                          SECERR:
                                ;
                                ;;*****************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
046044  013737  001376  051704          MOV     RMCS10,5155     ;STORE FUNCTION CODE
046052  042737  177701  051704          BIC     #^C<F0!F1!F2!F3!F4>,5155
046060  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
046064  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
046070  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
046074  032737  000200  001340          BIT     #DRY,RMDSI      ;DRIVE READY??
046102  001024                          BNE     5$              ;YES!!
046104  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
046112  042737  177577  001142          BIC     #^CDRY,$BDDAT
046120  012737  000200  001140          MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
046126  062716  000004                  ADD     #4,(SP)
046132  112776  000010  000000          MOVB    #10,@(SP)       ;ERROR NUMBER
046140  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
046144  004736                          JSR     PC,@(SP)+       ;REPORT NOT READY
046146  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
046152  000240                          NOP
                                ;
                                ;REPORT ERROR IF GO BIT IS NOT RESET
046154  032737  000001  001326  5$:     BIT     #GO,RMCS1I      ;GO BIT RESET??
046162  001423                          BEQ     10$             ;YES!!
046164  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
046172  042737  177776  001142          BIC     #^CGO,$BDDAT
046200  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
046204  062716  000004                  ADD     #4,(SP)
```

```
 9944  046210  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
 9945  046216  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9946  046222  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
 9947  046224  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9948  046230  000240                          NOP
 9949
 9950                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
 9951  046232  013737  001326  001142  10$:    MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
 9952  046240  042737  177701  001142          BIC     #^C76,$BDDAT
 9953  046246  013737  051704  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
 9954  046254  023737  001142  001140          CMP     $BDDAT,$GDDAT
 9955  046262  001413                          BEQ     15$             ;YES!!
 9956  046264  062716  000004                  ADD     #4,(SP)
 9957  046270  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
 9958  046276  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9959  046302  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
 9960  046304  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9961  046310  000240                          NOP
 9962  046312                          15$:
 9963                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
 9964                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
 9965                                  ;OTHER ERRORS ARE SET
 9966  046312  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
 9967  046316  005737  001342                  TST     RMER1I          ;IS RMER1 =0??
 9968  046322  001003                          BNE     20$             ;NO!!
 9969  046324  005737  001370                  TST     RMER2I          ;IS RMERZ=0??
 9970  046330  001403                          BEQ     25$             ;YES!!
 9971  046332  052737  040000  001140  20$:    BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
 9972  046340  013737  001340  001142  25$:    MOV     RMDS1,$BDDAT
 9973  046346  042737  137777  001142          BIC     #^CERR,$BDDAT
 9974  046354  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
 9975  046362  001412                          BEQ     30$             ;YES!!
 9976  046364  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9977  046370  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
 9978  046376  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9979  046402  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
 9980  046404  162716  000010                  SUB     #10,(SP)
 9981
 9982                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
 9983                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
 9984                                  ;SET TRE IS SET
 9985  046410  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
 9986  046414  013746  001336                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
 9987  046420  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
 9988  046424  001010                          BNE     35$             ;YES!!
 9989  046426  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
 9990  046434  001407                          BEQ     40$             ;NO!!
 9991  046436  022737  000030  051704          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9992  046444  103003                          BHIS    40$             ;NO!!
 9993  046446  052737  040000  001140  35$:    BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
 9994  046454  013737  001326  001142  40$:    MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
 9995  046462  042737  137777  001142          BIC     #^CTRE,$BDDAT
 9996  046470  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
 9997  046476  001413                          BEQ     45$             ;YES!!
 9998  046500  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9999  046504  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
10000  046512  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10001  046516  004736                  JSR     PC,@(SP)+       ;REPORT TRE ERROR
10002  046520  162716  000010          SUB     #10,(SP)        ;RESTORE (SP)
10003  046524  000240                  NOP
10004  046526                  45$:
10005
10006                          ;;*****************************************************************
10007                          ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
10008                          ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
10009                          ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
10010                          ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
10011                          ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
10012                          ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
10013
10014                          ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
10015  046526  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10016  046530  013700  051704          MOV     51$$,R0         ;R0 = FUNCTION CODE
10017  046534  016037  067254  051676  MOV     FNCDTB(R0),500$ ;STORE ENTRY
10018  046542  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
10019
10020                          ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
10021                          ;ATA IS NOT SET AND SHOULD BE SET.
10022  046544  013737  051676  001140  MOV     500$,$GDDAT     ;GET EXPECTED ATA STATUS
10023  046552  032737  040000  001340  BIT     #ERR,RMDSI      ;IS COMPOSITE ERROR SET ??
10024  046560  001403                  BEQ     50$             ;NO !!
10025  046562  052737  100000  001140  BIS     #ATA,$GDDAT     ;EXPECT AN ATTENTION
10026  046570  042737  077777  001140  50$:    BIC     #^CATA,$GDDAT   ;STRIP DONT CARES
10027  046576  013737  001340  001142  MOV     RMDSI,$BDDAT    ;GET RECEIVED ATA
10028  046604  042737  077777  001142  BIC     #^CATA,$BDDAT   ;STRIP DONT CARES
10029  046612  023737  001140  001142  CMP     $GDDAT,$BDDAT   ;IS ATA OK ??
10030  046620  001413                  BEQ     55$             ;YES !!
10031  046622  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10032  046626  112776  000006  000000  MOVB    #6,@(SP)        ;LOAD ERROR # IN CALL
10033  046634  162716  000002          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10034  046640  004736                  JSR     PC,@(SP)+       ;REPORT ERROR
10035  046642  162716  000010          SUB     #10,(SP)        ;RESTORE SP
10036  046646  000240                  NOP
10037  046650                  55$:
10038
10039                          ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
10040                          ;WITH FUNCTION CODE TABLE
10041  046650  013737  051676  001140          MOV     500$,$GDDAT     ;GET EXPECTED ILF
10042  046656  042737  177776  001140          BIC     #^CILF,$GDDAT   ;CLEAR ALL OTHER BITS
10043  046664  013737  001342  001142          MOV     RMER1I,$BDDAT   ;GET RECEIVED ILF
10044  046672  042737  177776  001142          BIC     #^CILF,$BDDAT   ;CLEAR ALL OTHER BITS
10045  046700  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ILF OK ??
10046  046706  001412                          BEQ     60$             ;YES !!
10047  046710  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10048  046714  112776  000254  000000          MOVB    #254,@(SP)      ;WRITE ERROR NUMBER IN CALL
10049  046722  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10050  046726  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10051  046730  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10052  046734  005037  001140          60$:    CLR     $GDDAT          ;CLEAR EXPECTED STATUS
10053
10054                          ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
10055  046740  013746  051676          MOV     500$,-(SP)      ;GET WCE STATUS ENABLE
```

```
10056  046744  052716  137777                  BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
10057  046750  013737  001336  001142          MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10058  046756  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
10059  046762  001412                          BEQ     90$             ;BRANCH IF WCE OK
10060  046764  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10061  046770  112776  000026  000000          MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
10062  046776  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10063  047002  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10064  047004  162716  000010                  SUB     #10,(SP)        ;RESTORE ERROR
10065  047010                          90$:
10066
10067                                  ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
10068  047010  013746  051676                  MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
10069  047014  052716  157777                  BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
10070  047020  013737  001342  001142          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10071  047026  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
10072  047032  001412                          BEQ     100$            ;BRANCH IF OPI OK
10073  047034  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10074  047040  112776  000164  000000          MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
10075  047046  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10076  047052  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10077  047054  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10078  047060                          100$:
10079
10080                                  ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
10081                                  ;SET AND VV IS NOT RESET
10082  047060  013746  051676                  MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
10083  047064  032737  000100  001340          BIT     #VV,RMDSI       ;IS VV SET
10084  047072  001402                          BEQ     105$            ;NO !!
10085  047074  042716  010000                  BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
10086  047100  052716  167777          105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
10087  047104  013737  001370  001142          MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10088  047112  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
10089  047116  001412                          BEQ     110$            ;BRANCH IF IVC OK
10090  047120  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10091  047124  112776  000165  000000          MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
10092  047132  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10093  047136  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10094  047140  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10095  047144                          110$:
10096
10097                                  ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10098                                  ; ALL WRITE ERRORS, I.E.,
10099                                  ;       RMER1 - WLE, WCF
10100                                  ;       RMER2 - DPE
10101                                  ;       RMCS2 - UPE.
10102                                  ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
10103                                  ;WRITE ERROR ENABLE BIT IS RESET.
10104
10105                                  ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
10106                                  ;THE DRIVE IS NOT WRITE PROTECTED
10107  047144  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10108  047150  032737  004000  051676          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10109  047156  001404                          BEQ     115$            ;NO !!
10110  047160  032737  004000  001340          BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
10111  047166  001002                          BNE     120$            ;YES !!
```

```
10112  047170  042716  004000            115$:   BIC     #WLE,(SP)          ;RESET WLE ENABLE
10113  047174  013737  001342  001142    120$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
10114  047202  042637  001142                    BIC     (SP)+,$BDDAT       ;CLEAR WLE IF ENABLED
10115  047206  001412                            BEQ     125$               ;BRANCH IF WLE OK
10116  047210  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10117  047214  112776  000023  000000            MOVB    #23,@(SP)          ;WRITE ERROR NUMBER IN CALL
10118  047222  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10119  047226  004736                            JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
10120  047230  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10121  047234                            125$:
10122
10123                                    ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
10124  047234  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ENABLED
10125  047240  032737  004000  051676            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10126  047246  001002                            BNE     130$               ;YES !!
10127  047250  042716  000040                    BIC     #WCF,(SP)          ;DISABLE WCF ERROR
10128  047254  013737  001342  001142    130$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
10129  047262  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET WCF IF ENABLED
10130  047266  001412                            BEQ     135$               ;BRANCH IF WCF OK
10131  047270  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10132  047274  112776  000025  000000            MOVB    #25,@(SP)          ;WRITE ERROR NUMBER IN CALL
10133  047302  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10134  047306  004736                            JSR     PC,@(SP)+          ;REPORT ERROR
10135  047310  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10136  047314                            135$:
10137
10138                                    ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10139  047314  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
10140  047320  032737  004000  051676            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10141  047326  001002                            BNE     140$               ;YES !!
10142  047330  042716  000010                    BIC     #DPE,(SP)          ;RESET DPE ENABLE
10143  047334  013737  001370  001142    140$:   MOV     RMER2I,$BDDAT      ;GET RECEIVED STATUS
10144  047342  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET DPE IF ENABLED
10145  047346  001412                            BEQ     145$               ;BRANCH IF DPE OK
10146  047350  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10147  047354  112776  000040  000000            MOVB    #40,@(SP)          ;WRITE ERROR NUMBER IN CALL
10148  047362  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10149  047366  004736                            JSR     PC,@(SP)+          ;REPORT ERROR
10150  047370  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10151  047374                            145$:
10152
10153                                    ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10154  047374  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
10155  047400  032737  004000  051676            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10156  047406  001002                            BNE     150$               ;YES !!
10157  047410  042716  020000                    BIC     #UPE,(SP)          ;DISABLE UPE ERROR
10158  047414  013737  001336  001142    150$:   MOV     RMCS2I,$BDDAT      ;GET RECEIVED STATUS
10159  047422  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET UPE IF ENABLED
10160  047426  001412                            BEQ     155$               ;BRANCH IF UPE OK
10161  047430  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10162  047434  112776  000024  000000            MOVB    #24,@(SP)          ;WRITE ERROR NUMBER IN CALL
10163  047442  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10164  047446  004736                            JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
10165  047450  162716  000010                    SUB     #10,(SP)           ;MOVE SP TO NO ERROR
10166  047454                            155$:
10167
```

```
10168                                   ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
10169  047454  013746  051676                    MOV     500$,-(SP)      ;GET IAE ENABLE
10170  047460  052716  175777                    BIS     #^C<IAE>,(SP)   ;SET ALL OTHER BITS
10171  047464  013737  001342  001142            MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10172  047472  042637  001142                    BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
10173  047476  001412                            BEQ     160$            ;BRANCH IF IAE IS OK
10174  047500  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10175  047504  112776  000166  000000            MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
10176  047512  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10177  047516  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10178  047520  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10179  047524                          160$:
10180
10181                                   ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10182                                   ;  ALL READ/WRITE ERRORS, I.E.,
10183                                   ;       RMCS1 - TRE
10184                                   ;       RMCS2 - DLT,NEM,MXF
10185                                   ;       RMDS  - LBT
10186                                   ;       RMER1 - AOE
10187                                   ;NOTE:
10188                                   ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
10189                                   ;CYLINDER REGISTER IS WRITTEN
10190                                   ;NOTE:
10191                                   ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
10192                                   ;NOTE:
10193                                   ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
10194
10195
10196                                   ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10197  047524  012746  177777                    MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10198  047530  032737  001000  051676            BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10199  047536  001002                            BNE     165$            ;YES !!
10200  047540  042716  100000                    BIC     #DLT,(SP)       ;RESET DLT ENABLE
10201  047544  013737  001336  001142  165$:     MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10202  047552  042637  001142                    BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
10203  047556  001412                            BEQ     170$            ;BRANCH IF DLT IS OK
10204  047560  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10205  047564  112776  000032  000000            MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
10206  047572  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10207  047576  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10208  047600  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10209  047604                          170$:
10210
10211                                   ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10212  047604  012746  177777                    MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10213  047610  032737  001000  051676            BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10214  047616  001002                            BNE     175$            ;YES !!
10215  047620  042716  004000                    BIC     #NEM,(SP)       ;DISABLE NEM
10216  047624  013737  001336  001142  175$:     MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10217  047632  042637  001142                    BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
10218  047636  001412                            BEQ     180$            ;BRANCH IF NEM IS OK
10219  047640  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10220  047644  112776  000167  000000            MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
10221  047652  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10222  047656  004736                            JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10223  047660  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
10224  047664                                  180$:
10225
10226                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10227  047664  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10228  047670  032737  001000  051676                  BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
10229  047676  001002                                  BNE     185$            ;YES !!
10230  047700  042716  001000                          BIC     #MXF,(SP)       ;DISABLE MXF ERROR
10231  047704  013737  001336  001142  185$:          MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10232  047712  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
10233  047716  001412                                  BEQ     190$            ;BRANCH IF MXF IS OK
10234  047720  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10235  047724  112776  000033  000000                  MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
10236  047732  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10237  047736  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10238  047740  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10239  047744                                  190$:
10240
10241                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
10242  047744  012746  177777                          MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
10243  047750  032737  001000  051676                  BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
10244  047756  001404                                  BEQ     191$            ;NO !!
10245  047760  032737  002000  001340                  BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
10246  047766  001002                                  BNE     195$            ;YES !!
10247  047770  042716  001000                  191$:   BIC     #AOE,(SP)       ;DISABLE AOE
10248  047774  013737  001342  001142  195$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10249  050002  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
10250  050006  001412                                  BEQ     200$            ;BRANCH IF AOE IS OK
10251  050010  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10252  050014  112776  000020  000000                  MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
10253  050022  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10254  050026  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10255  050030  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10256  050034                                  200$:
10257
10258                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10259                                          ;HEADER ERRORS, I.E.,
10260                                          ;       RMER1 - HCRC,HCE,FER
10261                                          ;       RMER2 - BSE
10262
10263                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
10264  050034  032737  002000  001360                  BIT     #HCI,RMOFI      ;IS HCI SET ??
10265  050042  001403                                  BEQ     201$            ;NO !!
10266  050044  042737  000200  051676                  BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
10267  050052                                  201$:
10268
10269                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
10270  050052  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10271  050056  032737  000200  051676                  BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10272  050064  001002                                  BNE     205$            ;YES !!
10273  050066  042716  000400                          BIC     #HCRC,(SP)      ;DISABLE HCRC
10274  050072  013737  001342  001142  205$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10275  050100  042637  001142                          BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
10276  050104  001412                                  BEQ     210$            ;BRANCH IF HCRC IS OK
10277  050106  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10278  050112  112776  000035  000000                  MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
10279  050120  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
10280  050124  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10281  050126  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10282  050132                          210$:
10283
10284                                  ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
10285  050132  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10286  050136  032737  000200  051676          BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
10287  050144  001002                          BNE     215$            ;YES !!
10288  050146  042716  000200                  BIC     #HCE,(SP)       ;DISABLE HCE
10289  050152  013737  001342  001142  215$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10290  050160  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
10291  050164  001412                          BEQ     220$            ;BRANCH IF HCE IS OK
10292  050166  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10293  050172  112776  000036  000000          MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
10294  050200  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10295  050204  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10296  050206  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10297  050212                          220$:
10298
10299                                  ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
10300  050212  012746  177777                  MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
10301  050216  032737  000200  051676          BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10302  050224  001002                          BNE     225$            ;YES !!
10303  050226  042716  000020                  BIC     #FER,(SP)       ;DISABLE FER
10304  050232  013737  001342  001142  225$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10305  050240  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
10306  050244  001412                          BEQ     230$            ;BRANCH IF FER OK
10307  050246  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10308  050252  112776  000037  000000          MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
10309  050260  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10310  050264  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10311  050266  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10312  050272                          230$:
10313
10314                                  ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
10315  050272  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10316  050276  032737  000200  051676          BIT     #HCE,500$       ;ARE THEY ENABLED ??
10317  050304  001002                          BNE     235$            ;YES !!
10318  050306  042716  100000                  BIC     #BSE,(SP)       ;DISABLE BSE
10319  050312  013737  001370  001142  235$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10320  050320  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
10321  050324  001412                          BEQ     240$            ;BRANCH IF BSE OK
10322  050326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10323  050332  112776  000354  000000          MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
10324  050340  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10325  050344  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10326  050346  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10327  050352                          240$:
10328
10329                                  ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
10330                                  ;FIELD ERRORS, I.E.,
10331                                  ;       RMCS2 - MDPE
10332                                  ;       RMER1 - DCK,ECH
10333                                  ;NOTE:
10334                                  ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
10335                                  ;DCK IS SET.
```

```
10336
10337                                           ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
10338  050352  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10339  050356  032737  000100  051676                  BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
10340  050364  001002                                  BNE     245$            ;YES !!
10341  050366  042716  000400                          BIC     #MDPE,(SP)      ;DISABLE MDPE
10342  050372  013737  001336  001142  245$:          MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10343  050400  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
10344  050404  001412                                  BEQ     250$            ;BRANCH IF MDPE OK
10345  050406  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10346  050412  112776  000027  000000                  MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
10347  050420  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10348  050424  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10349  050426  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10350  050432                                  250$:
10351
10352                                           ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
10353  050432  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10354  050436  032737  000100  051676                  BIT     #ECH,500$       ;ARE THEY ENABLED ??
10355  050444  001002                                  BNE     255$            ;YES !!
10356  050446  042716  100000                          BIC     #DCK,(SP)       ;DISABLE DCK
10357  050452  013737  001342  001142  255$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10358  050460  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
10359  050464  001412                                  BEQ     260$            ;BRANCH IF DCK IS OK
10360  050466  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10361  050472  112776  000030  000000                  MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
10362  050500  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10363  050504  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10364  050506  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10365  050512                                  260$:
;10366
10367                                           ;REPORT ERROR IF ECH IS SET AND,
10368                                           ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
10369                                           ;       ECI IS SET, OR
10370                                           ;       DCK IS NOT SET.
10371  050512  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ENABLED
10372  050516  032737  000100  051676                  BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
10373  050524  001410                                  BEQ     265$            ;NO !!
10374  050526  032737  004000  001360                  BIT     #ECI,RMOFI      ;IS ECI SET ??
10375  050534  001004                                  BNE     265$            ;YES !!
10376  050536  032737  100000  001342                  BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
10377  050544  001002                                  BNE     270$            ;YES !!
10378  050546  042716  000100                  265$:   BIC     #ECH,(SP)       ;DISABLE ECH
10379  050552  013737  001342  001142  270$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10380  050560  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
10381  050564  001412                                  BEQ     275$            ;BRANCH IF ECH IS OK
10382  050566  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10383  050572  112776  000031  000000                  MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
10384  050600  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10385  050604  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10386  050606  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10387  050612                                  275$:
10388
10389                                           ;;*****************************************************************
10390                                           ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
10391
```

```
10392  050612  022737  000030  051704          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10393  050620  103402                          BLO     280$            ;YES!!
10394  050622  000137  051650                  JMP     355$
10395
```

```
10396                                   ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
10397                                   ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10398  050626 013737 001330 001142 280$:  MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
10399  050634 001421                      BEQ     285$            ;YES
10400  050636 032737 040000 001326       BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
10401  050644 001015                      BNE     285$            ;YES!!
10402  050646 062716 000004               ADD     #4,(SP)
10403  050652 112776 000015 000000       MOVB    #15,@(SP)       ;ERROR NUMBER
10404  050660 005037 001140              CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
10405  050664 162716 000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10406  050670 004736                      JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
10407  050672 162716 000010              SUB     #10,(SP)        ;RESTORE (SP)
10408  050676 000240                      NOP
10409                                   ;REPORT ERROR IF RMBA IS NOT CORRECT
10410  050700 013737 001330 001140 285$:  MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
10411  050706 163737 001400 001140       SUB     RMWCO,$GDDAT
10412  050714 006337 001140              ASL     $GDDAT
10413  050720 063737 001402 001140       ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
10414  050726 032737 000010 001336       BIT     #BAI,RMCS2I     ;WAS BAI SET ??
10415  050734 001403                      BEQ     290$            ;NO !!
10416  050736 013737 001402 001140       MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
10417  050744 023737 001140 001332 290$:  CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
10418  050752 001416                      BEQ     295$            ;YES!!
10419  050754 013737 001332 001142       MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
10420  050762 062716 000004               ADD     #4,(SP)
10421  050766 112776 000016 000000       MOVB    #16,@(SP)       ;ERROR NUMBER
10422  050774 162716 000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10423  051000 004736                      JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
10424  051002 162716 000010              SUB     #10,(SP)                ;RESTORE (SP)
10425  051006 000240                      NOP
10426                                   ;COMPUTE NUMBER OF SECTORS TRANSFERRED
10427  051010 005046                   295$:  CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
10428  051012 013746 001330              MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
10429  051016 163716 001400              SUB     RMWCO,(SP)
10430  051022 012746 000400              MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
10431  051026 032737 000002 001376       BIT     #BIT1,RMCS1O    ;HEADER & DATA COMMAND ??
10432  051034 001402                      BEQ     300$            ;NO !!
10433  051036 012716 000402              MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
10434  051042 005266 000004           300$:  INC     4(SP)           ;INCREMENT SECTOR COUNT
10435  051046 161666 000002              SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
10436  051052 003373                      BGT     300$            ;CONTINUE IF NOT DONE
10437  051054 022626                      CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
10438                                   ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
10439  051056 013737 001404 051702       MOV     RMDAO,$10$      ;STORE ORIGINAL SECTOR
10440  051064 013737 001404 051700       MOV     RMDAO,$05$      ;STORE ORIGINAL TRACK
10441  051072 013737 001432 051676       MOV     RMDCO,$00$      ;STORE ORIGINAL CYLINDER
10442  051100 042737 177740 051702       BIC     #^C37,$10$
10443  051106 000337 051700              SWAB    $05$
10444  051112 042737 177770 051700       BIC     #^C7,$05$
10445  051120 062637 051702              ADD     (SP)+,$10$
10446
10447  051124 023727 051702 000040 305$:  CMP     $10$,#32.       ;SECTOR OVEFLOWED??
10448  051132 103420                      BLO     315$            ;NO!!
10449  051134 023727 051702 000240       CMP     $10$,#<5*32.>   ;CYLINDERS WORTH??
10450  051142 103406                      BLO     310$            ;NO!!
10451  051144 005237 051676              INC     $00$            ;YES INCREMENT CYLINDER
```

```
10452  051150  162737  000240  051702          SUB     #<5*32.>,510$   ;ADJUST SECTOR
10453  051156  000762                          BR      305$            ;TRY AGAIN
10454  051160  005237  051700          310$:   INC     505$            ;INCREMENT TRACK
10455  051164  162737  000040  051702          SUB     #32.,510$       ;ADJUST SECTOR
10456  051172  000754                          BR      305$            ;TRY AGAIN
10457
10458  051174  023727  051700  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
10459  051202  103406                          BLO     320$            ;NO!!
10460  051204  005237  051676                  INC     500$            ;INCREMENT CYLINDER
10461  051210  162737  000005  051700          SUB     #5,505$ ;ADJUST TRACK
10462  051216  000766                          BR      315$            ;TRY AGAIN
10463                          ;REPORT ERROR IF "LBT" IS NOT CORRECT
10464  051220  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
10465  051224  022737  001467  051676          CMP     #823.,500$      ;SHOULD LBT BE SET??
10466  051232  101007                          BHI     325$            ;NO!!
10467  051234  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS IAE SET ??
10468  051242  0C1003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
10469  051244  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
10470  051252  013737  001340  001142  325$:   MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
10471  051260  042737  175777  001142          BIC     #^CLBT,$BDDAT
10472  051266  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
10473  051274  001413                          BEQ     330$            ;YES!!
10474  051276  062716  000004                  ADD     #4,(SP)
10475  051302  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
10476  051310  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10477  051314  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
10478  051316  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10479  051322  000240                          NOP
10480                          ;REPORT ERROR IF "AOE" IS INCORRECT
10481  051324  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
10482  051330  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
10483  051336  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
10484  051340  022737  001467  051676          CMP     #823.,500$      ;SHOULD AOE BE SET??
10485  051346  101025                          BHI     340$            ;NO!!
10486  051350  005737  051700                  TST     505$            ;MAYBE
10487  051354  001012                          BNE     335$            ;YES
10488  051356  005737  051702                  TST     510$
10489  051362  001007                          BNE     335$            ;YES !!
10490  051364  032737  000010  051704          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
10491  051372  001413                          BEQ     340$            ;NO !!
10492  051374  005737  001330                  TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
10493  051400  001410                          BEQ     340$            ;YES !!
10494  051402  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
10495  051410  005037  051700                  CLR     505$            ;CLEAR EXPECTED TRACK
10496  051414  012737  000001  051702          MOV     #1,510$ ;EXPECT SECTOR=1
10497  051422  013737  001342  001142  340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10498  051430  042737  176777  001142          BIC     #^CAOE,$BDDAT
10499  051436  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
10500  051444  001413                          BEQ     345$            ;YES!!
10501  051446  062716  000004                  ADD     #4,(SP)
10502  051452  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
10503  051460  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10504  051464  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
10505  051466  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10506  051472  000240                          NOP
10507                          ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
10508  051474  032737  002000  001342  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
10509  051502  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
10510  051504  013737  051700  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
10511  051512  000337  001140                  SWAB    $GDDAT
10512  051516  113737  051702  001140          MOVB    510$,$GDDAT
10513  051524  013737  001334  001142          MOV     RMDA1,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
10514  051532  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10515  051540  001413                          BEQ     350$            ;BRANCH IF EQUAL
10516  051542  062716  000004                  ADD     #4,(SP)
10517  051546  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
10518  051554  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10519  051560  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
10520  051562  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10521  051566  000240                          NOP
10522                          ;REPORT ERROR IF RMDC IS INCORRECT
10523  051570  013737  051676  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
10524  051576  042737  176000  001140          BIC     #^C1777,$GDDAT
10525  051604  013737  001362  001142          MOV     RMDC1,$BDDAT    ;SETUP RECEIVED CYLINDER
10526  051612  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
10527  051620  001413                          BEQ     355$            ;BRANCH IF EQUAL
10528  051622  062716  000004                  ADD     #4,(SP)
10529  051626  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
10530  051634  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10531  051640  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
10532  051642  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10533  051646  000240                          NOP
```

```
10534   051650  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10535   051654  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
10536   051660  001403                          BEQ     360$
10537   051662  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
10538   051666  000402                          BR      365$
10539   051670  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10540   051674  000207                  365$:   RTS     PC
10541
10542   051676  000000                  500$:   .WORD   0               ;CYLINDER
10543   051700  000000                  505$:   .WORD   0               ;TRACK
10544   051702  000000                  510$:   .WORD   0               ;SECTOR
10545   051704  000000                  515$:   .WORD   0               ;FUNCTION CODE
10546
10547
```

```
                                        .SBTTL  DEVICE SELECT SUBROUTINE

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

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

051706                                  DEVSEL:

                                        ;CLEAR USER'S ERROR CALL
051706  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
051712  105076  000000                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
051716  162716  000004                          SUB     #4,(SP)         ;MOVE SP BACK
                                        ;SAVE USER'S INFORMATION AND SETUP REGISTERS
051722  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
051726  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
051732  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
051734  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
051736  012737  052056  000004                  MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
051744  012737  000300  000006                  MOV     #PR6,ERRVEC+2
051752  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
051756  013701  001446                          MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER

                                        ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
051762  111160  000010                          MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
051766  016037  000000  001176                  MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
051774  016037  000010  001174                  MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
052002  032737  010000  001174                  BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
052010  001407                                  BEQ     10$             ;NO!!
052012  062766  000004  000010                  ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
052020  112776  000111  000010                  MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
052026  000422                                  BR      30$             ;
052030  032737  004000  001176  10$:           BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
052036  001021                                  BNE     35$             ;YES!!
052040  062766  000004  000010                  ADD     #4,10(SP)
052046  112776  000112  000010                  MOVB    #112,@10(SP)
052054  000407                                  BR      30$

052056                                  20$:

                                        ;HANDLE BUS TIMEOUT
052056' 022626                                  CMP     (SP)+,(SP)+     ;ADJUST SP
052060  062766  000004  000010                  ADD     #4,10(SP)
052066  112776  000113  000010                  MOVB    #113,@10(SP)
052074  162766  000002  000010  30$:           SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR

052102                                  35$:

                                        ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
052102  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
052104  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 10604  052106  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 10605  052112  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 10606  052116  000207                        RTS     PC              ;EXIT
```

```
10607                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
10608                                   ;
10609                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
10610                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
10611                                   ;
10612                                   ;
10613                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
10614                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
10615                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
10616                                   ;
10617                                   ;CALL:
10618                                   ;(1)    JSR     PC,SEKSTS
10619                                   ;       BR      ???             RETURN HERE IF NO ERROR
10620                                   ;       NOP                     RETURN HERE TO REPORT AN ERROP
10621                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10622                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10623                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10624                                   ;
10625   052120                          SEKSTS:
10626                                   ;
10627                                   ;CLEAR USER'S ERROR  CALL
10628   052120  000240                          NOP
10629   052122  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10630   052126  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
10631   052132  162716  000004                SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10632   052136  005037  053356                CLR     300$            ;CLEAR ERROR FLAGS
10633                                   ;
10634                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
10635                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
10636   052142  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
10637   052150  001424                          BEQ     1$              ;NO!!
10638   052152  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
10639   052160  001020                          BNE     1$              ;NOT SURE!!
10640                                   ;
10641                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
10642   052162  005037  001140                CLR     $GDDAT          ;EXPECTED STATUS
10643   052166  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10644   052174  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
10645   052200  112776  000050  000000        MOVB    #50,@(SP)       ;ERROR #50
10646   052206  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10647   052212  004736                        JSR     PC,@(SP)+
10648   052214  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
10649   052220  000437                        BR      3$              ;IAE SHOULD  BE ZERO
10650
10651
10652                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
10653                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
10654   052222  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
10655   052230  052737  040000  053356          BIS     #SKI,300$       ;SET SKI ERROR FLAG
10656   052236  023727  001432  001466          CMP     RMDC0,#822.     ;CYLINDER > 822??
10657   052244  101025                          BHI     3$              ;YES!!
10658   052246  042737  040000  053356          BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
10659   052254  123727  001404  000037          CMPB    RMDA0,#31.      ;SECTOR > 31??
10660   052262  101016                          BHI     3$              ;YES!!
10661   052264  123727  001404  000035          CMPB    RMDA0,#29.      ;SECTOR > 29 ??
10662   052272  101404                          BLOS    2$              ;NO!!
```

```
10663  052274  032737  010000  001430          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
10664  052302  001406                          BEQ     3$              ;YES!!
10665  052304  123727  001405  000004  2$:     CMPB    RMDAO+1,#4.     ;TRACK  >4??
10666  052312  101002                          BHI     3$              ;YES!!
10667  052314  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
10668
10669                                  ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
10670  052320  013737  001342  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
10671  052326  042737  175777  001142          BIC     #^CIAE,$BDDAT
10672  052334  023737  001140  001142          CMP     $GDDAT,$BDDAT
10673  052342  001004                          BNE     35$             ;IAE IN ERROR
10674  052344  042737  040000  053356          BIC     #SKI,300$       ;CLEAR SKI FLAG
10675  052352  000413                          BR      5$              ;GO CHECK NEXT ERROR
10676  052354                          35$:
10677                                  ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
10678  052354  062716  000004                  ADD     #4,(SP)
10679  052360  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
10680  052366  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10681  052372  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
10682  052374  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10683  052400  000240                          NOP
10684  052402                          5$:
10685
10686                                  ;REPORT ANY IVC ERROR AS
10687                                  ;       IVC ERROR WITH VOLUME VALID ZERO
10688                                  ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
10689  052402  032737  010000  001370          BIT     #IVC,RMER2I     ;IVC ERROR??
10690  052410  001427                          BEQ     52$             ;NO!!
10691  052412  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
10692  052416  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10693  052424  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10694  052430  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
10695  052436  032737  000100  001340          BIT     #VV,RMDSI
10696  052444  001403                          BEQ     51$
10697  052446  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
10698  052454  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10699  052460  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10700  052462  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10701  052466  000240                          NOP
10702
10703  052470  013737  001370  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10704  052476  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
10705  052504  013737  053356  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
10706  052512  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
10707  052520  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
10708
10709                                  ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
10710  052522  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
10711  052530  001032                          BNE     54$             ;YES !!
10712  052532  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10713  052536  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
10714  052544  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10715  052550  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10716  052552  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10717  052556  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
10718  052560                          53$:
```

```
10719
10720                                          ;REPORT ERROR IF SKI IS SET
10721  052560  032737  040000  001142                  BIT     #SKI,$BDDAT     ;IS SKI SET ??
10722  052566  001413                                  BEQ     54$             ;NO - SKI IS OK
10723  052570  062716  000004                          ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10724  052574  112776  000054  000000                  MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
10725  052602  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10726  052606  004736                                  JSR     PC,@(SP)+       ;REPORT SEEK ERROR
10727  052610  162716  000010                          SUB     #10,(SP)        ;RESTORE (SP)
10728  052614  000240                                  NOP
10729
10730                                          ;REPORT ANY DEVICE CHECK
10731  052616  032737  000200  001370  54$:           BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
10732  052624  001420                                  BEQ     6$              ;NO!!
10733  052626  005037  001140                          CLR     $GDDAT          ;EXPECTED STATUS
10734  052632  013737  001370  001142                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10735  052640  062716  000004                          ADD     #4,(SP)
10736  052644  112776  000055  000000                  MOVB    #55,@(SP)       ;ERROR  #55
10737  052652  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10738  052656  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10739  052660  162716  000010                          SUB     #10,(SP)        ;RESTORE SP
10740  052664  000240                                  NOP
10741
10742                                          ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
10743                                          ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
10744  052666  032737  020000  001342  6$:           BIT     #OPI,RMER1I     ;"OPI" ERROR??
10745  052674  001427                                  BEQ     8$              ;NO!!
10746  052676  005037  001140                          CLR     $GDDAT          ;EXPECTED STATUS
10747  052702  013737  001342  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10748  052710  062716  000004                          ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10749  052714  112776  000052  000000                  MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
10750  052722  032737  010000  001340                  BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
10751  052730  001403                                  BEQ     7$              ;NO!!
10752  052732  112776  000053  000000                  MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
10753  052740  162716  000002                  7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10754  052744  004736                                  JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
10755  052746  162716  000010                          SUB     #10,(SP)        ;RESTORE (SP)
10756  052752  000240                                  NOP
10757
10758                                          ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
10759  052754  013746  001340                  8$:     MOV     RMDSI,-(SP)
10760  052760  042716  047677                          BIC     #^C<ATA!PIP!MOL!VV>,(SP)
10761  052764  022726  110100                          CMP     #ATA!MOL!VV,(SP)+
10762  052770  001002                                  BNE     9$              ;ERROR IN RMDS
10763  052772  000137  053326                          JMP     14$             ;RMDS IS OK
10764
10765                                          ;REPORT ERROR IF MOL = 0 AND OPI = 0
10766  052776  032737  010000  001340          9$:     BIT     #MOL,RMDSI      ;IS MOL RESET??
10767  053004  001030                                  BNE     10$             ;NO - MOL IS SET
10768  053006  032737  020000  001342                  BIT     #OPI,RMER1I     ;WAS OPI SET
10769  053014  001024                                  BNE     10$             ;YES - DONT REPORT ERROR
10770  053016  013737  001340  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10771  053024  052737  010000  001140                  BIS     #MOL,$GDDAT
10772  053032  013737  001340  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10773  053040  062716  000004                          ADD     #4,(SP)
10774  053044  112776  000062  000000                  MOVB    #62,@(SP)
```

```
10775  053052  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10776  053056  004736                          JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10777  053060  162716  000010                  SUB     #10,(SP)
10778  053064  000240                          NOP
10779
10780                                  ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
10781  053066  032737  020000  001340  10$:    BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
10782  053074  001430                          BEQ     11$              ;NO!!
10783  053076  032737  040000  001370          BIT     #SKI,RMER2I      ;WAS "SKI"SET??
10784  053104  001024                          BNE     11$              ;YES-DONT REPORT PIP
10785  053106  013737  001340  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10786  053114  042737  020000  001142          BIC     #PIP,$BDDAT
10787  053122  013737  001340  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10788  053130  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10789  053134  112776  000056  000000          MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
10790  053142  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10791  053146  004736                          JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
10792  053150  162716  000010                  SUB     #10,(SP)                 ;RESTORE (SP)
10793  053154  000240                          NOP
10794
10795                                  ;REPORT AN ERROR IF "ATA" IS NOT SET
10796  053156  032737  100000  001340  11$:    BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
10797  053164  001024                          BNE     13$              ;YES!!
10798  053166  013737  001340  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10799  053174  052737  110600  001140          BIS     #ATA!MOL!DPR!DRY,$GDDAT
10800  053202  013737  001340  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10801  053210  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10802  053214  112776  000057  000000          MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
10803  053222  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10804  053226  004736                          JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
10805                                                                   ;SEEK TEST
10806  053230  162716  000010                  SUB     #10,(SP)                 ;RESTORE (SP)
10807  053234  000240                          NOP
10808
10809                                  ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
10810  053236  032737  000100  001340  13$:    BIT     #VV,RMDSI        ;IS VV = 0 ??
10811  053244  001030                          BNE     14$              ;NO!!
10812  053246  032737  010000  001370          BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
10813  053254  001024                          BNE     14$              ;NO - IVC IS SET
10814  053256  013737  001340  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10815  053264  052737  000100  001140          BIS     #VV,$GDDAT
10816  053272  013737  001340  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10817  053300  062716  000004                  ADD     #4,(SP)
10818  053304  112776  000064  000000          MOVB    #64,@(SP)        ;ERROR #64
10819  053312  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10820  053316  004736                          JSR     PC,@(SP)+
10821  053320  162716  000010                  SUB     #10,(SP)
10822  053324  000240                          NOP
10823  053326                          14$:
10824
10825                                  ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10826  053326  000240                          NOP
10827  053330  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
10828  053334  105776  000000                  TSTB    @(SP)            ;WAS ERROR CALLED??
10829  053340  001403                          BEQ     15$              ;NO!!
10830  053342  062716  000004                  ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
10831   053346  000402                          BR      16$
10832
10833   053350  162716  000004          15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10834   053354  000207                  16$:    RTS     PC              ;RETURN
10835
10836   053356  000000                  300$:   .WORD   0               ;ERROR FLAGS
10837
```

```
                                .SBTTL  CONTROLLER CLEAR SUBROUTINE

                                ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
                                ;AND DRIVES, THEN SELECTS THE DRIVE.
                                ;
                                ;CALL:  JSR     PC,CNTCLR
                                ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;       ERROR                   SUB DEFINES ERROR NUMBER
                                ;       ???
                                ;
                                CNTCLR:
053360                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
053360  010046                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
053362  010146                         MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
053364  013746  000004                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
053370  013746  000006                 MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
053374  012737  053434  000004         MOV     #PR6,ERRVEC+2
053402  012737  000300  000006         MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
053410  013700  001276                 MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
053414  012760  000040  000010         MOV     TSTQUE,R1
053422  013701  001446                 MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
053426  111160  000010                 BR      20$
053432  000412                 10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
053434  022626                         ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
053436  062766  000004  000010         MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
053444  112776  000007  000010         SUB     #2,10(SP)
053452  162766  000002  000010 20$:
053460                                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
053460  012637  000006                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
053464  012637  000004                 MOV     (SP)+,R1        ;;POP STACK INTO R1
053470  012601                         MOV     (SP)+,R0        ;;POP STACK INTO R0
053472  012600                         RTS     PC
053474  000207
10871
```

```
10872                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
10873
10874                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
10875                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
10876                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
10877                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
10878
10879                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
10880                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10881                                   ;
10882                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
10883                                   ;
10884
10885                                   ;CALL:
10886                                   ;(1)    JSR     PC,CLRSTS
10887                                   ;       BR      ???             RETURN HERE IF NO ERROR
10888                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10889                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10890                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10891                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10892
10893   053476                          CLRSTS:
10894
10895                                   ;CLEAR USER'S ERROR CALL
10896   053476  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR
10897   053502  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
10898   053506  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
10899                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10900   053512  013737  001326  001142 4$:     MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
10901   053520  042737  100000  001142         BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
10902   053526  012737  004200  001140         MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
10903   053534  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
10904   053542  001413                         BEQ     5$              ;BRANCH IF EQUAL
10905   053544  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10906   053550  112776  000126  000000         MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
10907   053556  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10908   053562  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10909   053564  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10910   053570  000240                         NOP
10911                                   ;REPORT ERROR IF RMBA NOT RESET
10912   053572  005037  001140 5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
10913   053576  013737  001332  001142         MOV     RMBAI,$BDDAT
10914   053604  001413                         BEQ     7$              ;BRANCH IF ZERO
10915   053606  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10916   053612  112776  000127  000000         MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
10917   053620  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10918   053624  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10919   053626  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10920   053632  000240                         NOP
10921                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
10922   053634  013737  001336  001142 7$:     MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
10923   053642  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10924   053644  005046                         CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
10925   053646  013701  001446                 MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
10926   053652  111116                         MOVB    (R1),(SP)
10927   053654  052716  000100                 BIS     #IR,(SP)
```

```
10928  053660  012637  001140                  MOV     (SP)+,$GDDAT
10929  053664  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10930  053666  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10931  053674  001413                          BEQ     9$              ;BRANCH IF EQUAL
10932  053676  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10933  053702  112776  000130  000000          MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
10934  053710  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10935  053714  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10936  053716  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10937  053722  000240                          NOP
10938                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
10939  053724  005037  001140          9$:     CLR     $GDDAT          ;VERIFY RMER1
10940  053730  013737  001342  001142          MOV     RMER1I,$BDDAT
10941  053736  042737  040000  001142          BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
10942  053744  001413                          BEQ     13$             ;BRANCH IF ZERO
10943  053746  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10944  053752  112776  000131  000000          MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
10945  053760  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10946  053764  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10947  053766  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10948  053772  000240                          NOP
10949                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
10950  053774  013737  001352  001142  13$:     MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
10951  054002  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
10952  054010  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
10953  054016  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10954  054024  001413                          BEQ     17$             ;BRANCH IF 0
10955  054026  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10956  054032  112776  000133  000000          MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
10957  054040  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10958  054044  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10959  054046  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10960  054052  000240                          NOP
10961                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
10962  054054  005037  001140          17$:    CLR     $GDDAT          ;EXPECT ZEROS
10963  054060  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
10964  054066  001413                          BEQ     19$
10965  054070  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10966  054074  112776  000135  000000          MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
10967  054102  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10968  054106  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10969  054110  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10970  054114  000240                          NOP
10971                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
10972  054116  013737  001366  001142  19$:     MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
10973  054124  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
10974  054132  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10975  054140  023737  001140  001142          CMP     $GDDAT,$BDDAT
10976  054146  001413                          BEQ     21$
10977  054150  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10978  054154  112776  000136  000000          MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
10979  054162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10980  054166  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10981  054170  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10982  054174  000240                          NOP
10983                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
10984  054176  005037  001140                21$:    CLR     $GDDAT                  ;EXPECT ALL ZEROS
10985  054202  013737  001370  001142                MOV     RMER2I,$BDDAT           ;VERIFY RMER2
10986  054210  042737  040200  001142                BIC     #SKI!DVC,$BDDAT         ;IGNORE DEVICE ERRORS
10987  054216  001413                                BEQ     215$                    ;BRANCH IF OTHER BITS 0
10988  054220  062716  000004                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
10989  054224  112776  000174  000000                MOVB    #174,@(SP)              ;WITE ERROR NUMBER IN CALL
10990  054232  162716  000002                        SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
10991  054236  004736                                JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
10992  054240  162716  000010                        SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
10993  054244  000240                                NOP
10994                                          ;REPORT ERROR IF RMDS NOT INITIALIZED
10995  054246  013737  001340  001142        215$:   MOV     RMDSI,$BDDAT            ;TEST DRIVE STATUS REGISTER
10996  054254  042737  177177  001142                BIC     #^C<DRY!DPR>,$BDDAT
10997  054262  012737  000600  001140                MOV     #DPR!DRY,$GDDAT         ;EXPECTED DRIVE STATUS
10998  054270  023737  001140  001142                CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED & RECEIVED
10999  054276  001413                                BEQ     22$                     ;BRANCH IF EQUAL
11000  054300  062716  000004                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11001  054304  112776  000134  000000                MOVB    #134,@(SP)              ;WRITE ERROR NUMBER
11002  054312  162716  000002                        SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
11003  054316  004736                                JSR     PC,@(SP)+               ;REPORT ERROR TO USER
11004  054320  162716  000010                        SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
11005  054324  000240                                NOP
11006  054326  062716  000004                22$:    ADD     #4,(SP)                 ;MOVE SP TO ERROR CALL
11007  054332  105776  000000                        TSTB    @(SP)                   ;WAS AN ERROE DETECTED??
11008  054336  001403                                BEQ     23$                     ;NO!!
11009  054340  062716  000004                        ADD     #4,(SP)                 ;YES - MOVE TO ERROR RETURN
11010  054344  000402                                BR      24$
11011  054346  162716  000004                23$:    SUB     #4,(SP)                 ;MOVE SP TO NO ERROR RETURN
11012  054352  000240                         24$:    NOP
11013  054354  000207                                RTS     PC
```

```
                                        .SBTTL  PACK ACKNOWLEDGE STATUS CHECK

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

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

054356                                  ACKSTS:

                                        ;CLEAR USER'S ERROR CALL
054356  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
054362  105076  000000                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
054366  162716  000004                          SUB     #4,(SP)         ;MOVE SP BACK

                                        ;REPORT AN ERROR  IF "VV" IS 0
054372  032737  000100  001340                  BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
054400  001024                                  BNE     1$              ;YES!!
054402  013737  001340  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
054410  052737  000100  001140                  BIS     #VV,$GDDAT
054416  013737  001340  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
054424  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
054430  112776  000155  000000                  MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
054436  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054442  004736                                  JSR     PC,@(SP)+       ;REPORT THE ERROR
054444  162716  000010                          SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054450  000240                                  NOP
054452                                  1$:

                                        ;REPORT AN ERROR IF "MOL" IS 0
054452  032737  010000  001340                  BIT     #MOL,RMDSI      ;IS MOL SET??
054460  001024                                  BNE     2$              ;YES!!
054462  013737  001340  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
054470  052737  010000  001140                  BIS     #MOL,$GDDAT
054476  013737  001340  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
054504  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
054510  112776  000041  000000                  MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
054516  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054522  004736                                  JSR     PC,@(SP)+       ;REPORT TH ERROR
054524  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO BRANCH
054530  000240                                  NOP
054532                                  2$:

                                        ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
054532  032737  060007  001342                  BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
054540  001570                                  BEQ     7$

                                        ;REPORT AN ERROR IF "UNS" IS SET
054542  032737  040000  001342                  BIT     #UNS,RMER1I     ;WAS UNS SET??
054550  001424                                  BEQ     3$              ;NO!!
```

```
11070  054552  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11071  054560  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11072  054566  042737  040000  001140          BIC     #UNS,$GDDAT
11073  054574  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11074  054600  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11075  054606  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11076  054612  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11077  054614  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11078  054620  000240                          NOP
11079  054622                          3$:
11080
11081                                  ;REPORT ANY OPI ERROR
11082  054622  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET ??
11083  054630  001424                          BEQ     4$              ;NO!!
11084  054632  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11085  054640  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11086  054646  042737  020000  001140          BIC     #OPI,$GDDAT
11087  054654  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11088  054660  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11089  054666  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11090  054672  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11091  054674  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11092  054700  000240                          NOP
11093  054702                          4$:
11094
11095                                  ;REPORT ANY RMR ERROR
11096  054702  032737  000004  001342          BIT     #RMR,RMER1I     ;WAS RMR SET??
11097  054710  001424                          BEQ     5$              ;NO!!
11098  054712  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11099  054720  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11100  054726  042737  000004  001140          BIC     #RMR,$GDDAT
11101  054734  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11102  054740  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11103  054746  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11104  054752  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11105  054754  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11106  054760  000240                          NOP
11107  054762                          5$:
11108
11109                                  ;REPORT ANY     ILR ERROR
11110  054762  032737  000002  001342          BIT     #ILR,RMER1I     ;WAS ILR SET??
11111  054770  001424                          BEQ     6$              ;NO!!
11112  054772  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11113  055000  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11114  055006  042737  000002  001140          BIC     #ILR,$GDDAT
11115  055014  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11116  055020  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11117  055026  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11118  055032  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11119  055034  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11120  055040  000240                          NOP
11121  055042                          6$:
11122
11123                                  ;REPORT ANY     ILF ERROR
11124  055042  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS ILF SET??
11125  055050  001424                          BEQ     7$              ;NO!!
```

```
11126  055052  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11127  055060  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11128  055066  042737  000001  001140          BIC     #ILF,$GDDAT
11129  055074  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11130  055100  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11131  055106  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11132  055112  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11133  055114  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11134  055120  000240                          NOP
11135  055122                          7$:
11136
11137                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
11138  055122  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11139  055126  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11140  055132  001403                          BEQ     8$              ;NO!!
11141  055134  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11142  055140  000402                          BR      9$
11143  055142  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11144  055146  000240                  9$:     NOP
11145  055150  000207                          RTS     PC
11146
```

```
11147                                   .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
11148
11149                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
11150                                   ;USING THE STATUS STORED IN THE GET BUFFER.
11151
11152                                   ;CALL:
11153                                   ;
11154                                   ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
11155                                   ;       BR      ???             RETURN HERE IF NO ERROR
11156                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11157                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11158                                   ;       JSR     PC,@(SF)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11159                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11160
11161   055152                          RCLSTS:
11162
11163                                   ;CLEAR USER'S ERROR NUMBER
11164   055152  062716  000004                ADD     #4,(SP)
11165   055156  105076  000000                CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
11166   055162  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
11167
11168
11169                                   ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
11170   055166  032737  022011  001342        BIT     #OPI!PAR!ILF!IAE,RMER1I
11171   055174  001553                        BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
11172
11173                                   ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
11174                                   ;"PAR" = 1 AND "DPE" = 0
11175   055176  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS "PAR" SET??
11176   055204  001430                        BEQ     1$              ;NO!!
11177   055206  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS "DPE" SET??
11178   055214  001024                        BNE     1$              ;YES - NOT A REGISTER ERROR
11179   055216  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11180   055224  042737  000010  001140        BIC     #PAR,$GDDAT
11181   055232  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11182   055240  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11183   055244  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
11184   055252  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11185   055256  004736                        JSR     PC,@(SP)+       ;GO REPORT ERROR
11186   055260  162716  000010                SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11187   055264  000240                        NOP
11188   055266                          1$:
11189
11190                                   ;REPORT ANY "ILF" ERROR
11191   055266  032737  000001  001342        BIT     #ILF,RMER1I     ;WAS "ILF" SET??
11192   055274  001424                        BEQ     2$              ;NO!!
11193   055276  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11194   055304  042737  000001  001140        BIC     #ILF,$GDDAT
11195   055312  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11196   055320  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11197   055324  112776  000071  000000        MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
11198   055332  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11199   055336  004736                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11200   055340  162716  000010                SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11201   055344  000240                        NOP
11202   055346                          2$:
```

```
11203
11204                                   ;REPORT ANY "OPI" ERROR AS
11205                                   ;       . OPI DUE TO "MOL" = 0
11206                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
11207  055346  032737  020000  001342           BIT     #OPI,RMER1I     ;WAS OPI SET??
11208  055354  001433                           BEQ     31$             ;NO!!
11209  055356  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11210  055364  042737  020000  001140           BIC     #OPI,$GDDAT
11211  055372  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11212  055400  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11213  055404  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11214  055412  032737  010000  001340           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
11215  055420  001403                           BEQ     3$              ;YES!!
11216  055422  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
11217  055430  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11218  055434  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11219  055436  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11220  055442  000240                           NOP
11221  055444                           31$:
11222
11223                                   ;REPORT AN ERROR IF "IAE" IS SET
11224  055444  032737  002000  001342           BIT     #IAE,RMER1I     ;IS "IAE" SET??
11225  055452  001424                           BEQ     4$              ;NO!!
11226  055454  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11227  055462  042737  002000  001140           BIC     #IAE,$GDDAT
11228  055470  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11229  055476  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11230  055502  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11231  055510  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11232  055514  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
11233  055516  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
11234  055522  000240                           NOP
11235  055524                           4$:
11236
11237                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
11238  055524  032737  050200  001370           BIT     #SKI!IVC!DVC,RMER2I
11239  055532  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
11240
11241
11242                                   ;REPORT ANY "IVC" ERROR AS
11243                                   ;       .IVC WITH VV = 0
11244                                   ;       .ERRONEOUS IVC ERROR
11245  055534  032737  010000  001370           BIT     #IVC,RMER2I     ;WAS IVC SET??
11246  055542  001433                           BEQ     6$              ;NO!!
11247  055544  013737  001370  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11248  055552  042737  010000  001140           BIC     #IVC,$GDDAT
11249  055560  013737  001370  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11250  055566  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11251  055572  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
11252  055600  032737  000100  001340           BIT     #VV,RMDSI       ;WAS VV = 0??
11253  055606  001403                           BEQ     5$              ;YES!!
11254  055610  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
11255  055616  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11256  055622  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11257  055624  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11258  055630  000240                           NOP
```

```
11259  055632                                  6$:
11260
11261                                          ;REPORT ANY "SKI" ERROR
11262  055632  032737  040000  001370                BIT     #SKI,RMER2I     ;WAS SKI SET??
11263  055640  001424                              BEQ     7$              ;NO!!
11264  055642  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11265  055650  042737  040000  001140              BIC     #SKI,$GDDAT
11266  055656  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11267  055664  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11268  055670  112776  000076  000000              MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
11269  055676  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11270  055702  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11271  055704  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO BRANCH
11272  055710  000240                              NOP
11273  055712                                  7$:
11274
11275                                          ;REPORT ANY "DVC" ERROR
11276  055712  032737  000200  001370                BIT     #DVC,RMER2I     ;WAS "DVC" SET??
11277  055720  001424                              BEQ     8$              ;NO!!
11278  055722  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11279  055730  042737  000200  001140              BIC     #DVC,$GDDAT
11280  055736  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11281  055744  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11282  055750  112776  000077  000000              MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
11283  055756  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11284  055762  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11285  055764  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11286  055770  000240                              NOP
11287  055772                                  8$:
11288
11289                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
11290  055772  013746  001340                      MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
11291  055776  042716  047676                      BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
11292  056002  022726  110100                      CMP     #ATA!MOL!VV,(SP)+
11293  056006  001002                              BNE     85$
11294  056010  000137  056424                      JMP     13$
11295  056014                                  85$:
11296
11297
11298                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
                                               ;LINE AFTER RECALIBRATE WAS INITIATED
11299  056014  032737  010000  001340                BIT     #MOL,RMDSI      ;DID MOL DROP??
11300  056022  001030                              BNE     9$              ;NO!!
11301  056024  032737  020000  001342              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11302  056032  001024                              BNE     9$              ;YES - DON'T REPORT MOL=0
11303  056034  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11304  056042  052737  010000  001140              BIS     #MOL,$GDDAT
11305  056050  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11306  056056  062716  000004                      ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11307  056062  112776  000100  000000              MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
11308  056070  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11309  056074  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11310  056076  162716  000010                      SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11311  056102  000240                              NOP
11312  056104                                  9$:
11313
11314                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
11315  056104  032737  000100  001340          BIT     #VV,RMDSI       ;DID "VV" DROP??
11316  056112  001030                          BNE     10$             ;NO!!
11317  056114  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
11318  056122  001024                          BNE     10$             ;YES - DONT REPORT VV=0
11319  056124  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11320  056132  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11321  056140  052737  000100  001140          BIS     #VV,$GDDAT
11322  056146  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11323  056152  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
11324  056160  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11325  056164  004736                          JSR     PC,@(SP)+
11326  056166  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11327  056172  000240                          NOP
11328  056174                          10$:
11329
11330                          ;REPORT AN ERROR IF ATA IS NOT SET
11331  056174  032737  100000  001340          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
11332  056202  001024                          BNE     11$             ;YES!!
11333  056204  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11334  056212  052737  100000  001140          BIS     #ATA,$GDDAT
11335  056220  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11336  056226  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11337  056232  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
11338  056240  162716  000002                  SUB     #2,(SP)
11339  056244  004736                          JSR     PC,@(SP)+
11340  056246  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11341  056252  000240                          NOP
11342
11343  056254                          11$:
11344
11345                          ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
11346                          ;ALWAYS CLEAR OFFSET MODE
11347  056254  032737  000001  001340          BIT     #OM,RMDSI       ;WAS "OM" RESET??
11348  056262  001424                          BEQ     12$             ;YES!!
11349  056264  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11350  056272  042737  000001  001140          BIC     #OM,$GDDAT
11351  056300  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11352  056306  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11353  056312  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
11354  056320  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11355  056324  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11356  056326  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11357  056332  000240                          NOP
11358  056334                          12$:
11359
11360                          ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
11361                          ;CYLINDER
11362  056334  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11363  056342  001430                          BEQ     13$             ;NO!!
11364  056344  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11365  056352  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
11366  056354  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11367  056362  042737  020000  001140          BIC     #PIP,$GDDAT
11368  056370  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11369  056376  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11370  056402  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
11371  056410  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
11372  056414  004736                          JSR     PC,@(SP)+
11373  056416  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO USER'S BRANCH
11374  056422  000240                          NOP
11375  056424                          13$:
11376
11377                                  ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
11378  056424  032737  040006  001342          BIT     #ILR!RMR!UNS,RMER1I
11379  056432  001514                          BEQ     16$
11380
11381                                  ;REPORT AN ERROR IF "ILR" IS SET
11382  056434  032737  000002  001342          BIT     #ILR,RMER1I         ;WAS ILR SET DURING RECALIBRATE??
11383  056442  001424                          BEQ     14$                 ;NO!!
11384  056444  013737  001342  001140          MOV     RMER1I,$GDDAT       ;EXPECTED STATUS
11385  056452  042737  000002  001140          BIC     #ILR,$GDDAT
11386  056460  013737  001342  001142          MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
11387  056466  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11388  056472  112776  000105  000000          MOVB    #105,@(SP)          ;WRITE ERROR NUMBER IN CALL
11389  056500  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
11390  056504  004736                          JSR     PC,@(SP)+
11391  056506  162716  000010                  SUB     #10,(SP)            ;MOVE SP TO USER'S BRANCH
11392  056512  000240                          NOP
11393  056514                          14$:
11394
11395                                  ;REPORT AN ERROR IF "RMR" IS SET
11396  056514  032737  000004  001342          BIT     #RMR,RMER1I         ;WAS RMR SET??
11397  056522  001424                          BEQ     15$                 ;NO!!
11398  056524  013737  001342  001140          MOV     RMER1I,$GDDAT       ;EXPECTED STATUS
11399  056532  042737  000004  001140          BIC     #RMR,$GDDAT
11400  056540  013737  001342  001142          MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
11401  056546  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
11402  056552  112776  000106  000000          MOVB    #106,@(SP)          ;WRITE ERROR NUMBER IN USER'S CALL
11403  056560  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
11404  056564  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
11405  056566  162716  000010                  SUB     #10,(SP)            ;MOVE SP TO USER'S BRANCH
11406  056572  000240                          NOP
11407  056574                          15$:
11408
11409                                  ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
11410  056574  032737  040000  001342          BIT     #UNS,RMER1I         ;WAS UNSAFE ON??
11411  056602  001430                          BEQ     16$                 ;NO!!
11412  056604  032737  000200  001370          BIT     #DVC,RMER2I         ;WAS THERE A DEVICE CHECK??
11413  056612  001024                          BNE     16$                 ;YES - DON'T REPORT UNSAFE
11414  056614  013737  001342  001140          MOV     RMER1I,$GDDAT       ;EXPECTED STATUS
11415  056622  042737  040000  001140          BIC     #UNS,$GDDAT
11416  056630  013737  001342  001142          MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
11417  056636  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
11418  056642  112776  000107  000000          MOVB    #107,@(SP)          ;WRITE ERROR NUMBER
11419  056650  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
11420  056654  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
11421  056656  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO USER'S BRANCH
11422  056662  000240                          NOP
11423  056664                          16$:
11424
11425                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11426  056664  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
```

```
11427  056670  105776  000000          TSTB    @(SP)           ;WAS AN ERROR REPORTED??
11428  056674  001403                  BEQ     17$             ;NO!!
11429  056676  062716  000004          ADD     #4,(SP)         ;YES - AUGMENT SP RETURN
11430  056702  000402                  BR      18$
11431  056704  162716  000004  17$:    SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
11432  056710  000240          18$:    NOP
11433  056712  000207                  RTS     PC              ;STATUS CECK IS COMPLETE
11434
```

```
                                        .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
                                        ;       BR      ???             RETURN HERE IF NO ERROR
                                        ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                        ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                        ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                        ;       ???                     RETURN HERE IF NO MORE ERRORS

056714                                  DRVSTS:

                                        ;CLEAR USER'S ERROR CALL
056714  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056720  105076  000000                          CLRB    @(SP)           ;CLEAR ERROR CALL
056724  162716  000004                          SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH

                                        ;REPORT ERROR IF RMCS1 NOT INITIALIZED
056730  013737  001326  001142        4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
056736  042737  173700  001142              BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
056744  012737  004010  001140              MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
056752  023737  001140  001142              CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
056760  001443                              BEQ     6$              ;BRANCH IF EQUAL
056762  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056766  112776  000141  000000              MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
056774  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
057000  004736                              JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
057002  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
057006  000240                              NOP

                                        ;REPORT ERROR IF RMDS NOT INITIALIZED
057010  013737  001340  001142        5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
057016  042737  021101  001142              BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
057024  012737  010600  001140              MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
057032  023737  001140  001142              CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
057040  001413                              BEQ     6$              ;BRANCH IF EQUAL
057042  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
057046  112776  000142  000000              MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
057054  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
057060  004736                              JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
057062  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
057066  000240                              NOP

                                        ;REPORT ERROR IF RMER1 NOT INITIALIZED
057070  005037  001140            6$:     CLR     $GDDAT          ;EXPECT 0'S
057074  013737  001342  001142              MOV     RMER1I,$BDDAT   ;CHECK RMER1
057102  001413                              BEQ     8$              ;BRANCH IF EQUAL
057104  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
057110  112776  000143  000000              MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
057116  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
057122  004736                              JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
057124  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
057130  000240                              NOP

                                        ;REPORT ERROR IF ATA NOT INITIALIZED
057132  013737  001344  001142        8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
057140  010146                              MOV     R1,-(SP)        ;;PUSH R1 ON STACK
057142  010246                              MOV     R2,-(SP)        ;;PUSH R2 ON STACK
057144  013701  001446                      MOV     TSTQUE,R1
057150  116102  000001                      MOVB    1(R1),R2
057154  042702  177400                      BIC     #^CATNMSK,R2
057160  005102                              COM     R2
057162  040237  001142                      BIC     R2,$BDDAT
```

```
11491  057166  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
11492  057170  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
11493  057172  005737  001142                TST     $BDDAT          ;IS ATTENTION CLEARED??
11494  057176  001413                        BEQ     9$              ;BRANCH IF ATTENTION CLEARED
11495  057200  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11496  057204  112776  000144  000000        MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11497  057212  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11498  057216  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11499  057220  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11500  057224  000240                        NOP
11501                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
11502  057226  013737  001352  001142  9$:   MOV     RMMR1I,$BDDAT   ;CHECK RMMR
11503  057234  042737  000046  001142        BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
11504  057242  012737  000010  001140        MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
11505  057250  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11506  057256  001413                        BEQ     11$             ;BRANCH IF ZERO
11507  057260  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11508  057264  112776  000145  000000        MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11509  057272  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11510  057276  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11511  057300  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11512  057304  000240                        NOP
11513                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
11514  057306  013737  001366  001142  11$:   MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
11515  057314  042737  140000  001142        BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
11516  057322  012737  011777  001140        MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
11517  057330  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11518  057336  001413                        BEQ     15$             ;BRANCH IF EQUAL
11519  057340  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11520  057344  112776  000146  000000        MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11521  057352  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11522  057356  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11523  057360  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11524  057364  000240                        NOP
11525  057366  005037  001140        15$:    CLR     $GDDAT          ;EXPECT ZEROS
11526                          ;REPORT ERROR IF RMEC2 NOT RESET
11527  057372  013737  001374  001142        MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
11528  057400  001413                        BEQ     17$             ;BRANCH IF 0
11529  057402  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11530  057406  112776  000150  000000        MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11531  057414  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11532  057420  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11533  057422  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11534  057426  000240                        NOP
11535                          ;REPORT ERROR IF RMER2 NOT RESET
11536  057430  013737  001370  001142  17$:   MOV     RMER2I,$BDDAT   ;CHECK RMER2
11537  057436  001413                        BEQ     18$             ;BRANCH IF NO ERROR
11538  057440  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11539  057444  112776  000147  000000        MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11540  057452  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11541  057456  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11542  057460  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11543  057464  000240                        NOP
11544  057466                          18$:
11545
11546  057466                          19$:
```

```
11547
11548                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
11549  057466  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11550  057472  105776  000000                TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11551  057476  001403                        BEQ     21$             ;NO!!
11552  057500  062716  000004                ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11553  057504  000402                        BR      23$
11554  057506  162716  000004        21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
11555  057512  000240                23$:    NOP
11556  057514  000207                        RTS     PC              ;RETURN TO USER
```

```
11557                                   .SBTTL  SEARCH STATUS CHECK SUBROUTINE
11558
11559                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEARCH OPERATIONS USING
11560                                   ;STATUS STORED IN THE GET BUFFER AND TEST CONDITIONS STORED IN THE
11561                                   ;PUT BUFFER.
11562
11563                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS
11564                                   ;DETECTED AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN
11565                                   ;THE USER'S "ERROR" TRAP.
11566
11567                                   ;THE FOLLOWING CONDITIONS ARE CHECKED:
11568                                   ;
11569                                   ;       .ANY ERROR WHICH OCCURRED WHILE READING OR WRITING REMOTE
11570                                   ;REGISTERS IS REPORTED, I.E., "MCPE"=1 OR "PAR"=1
11571                                   ;
11572                                   ;       ."IAE" STATUS IS CHECKED IN ACCORDANCE WITH THE VALUE DETERMINED
11573                                   ;BY THE PROGRAM, WHICH IS BASED ON FORMAT AND ADDRESS.
11574                                   ;
11575                                   ;       ."OPI", IF SET, IS REPORTED AS 1) "OPI" DUE TO MOL=0, OR 2)
11576                                   ;"OPI" DUE TO ON CYLINDER LATCH.
11577                                   ;
11578                                   ;       ."IVC", IF SET, IS REPORTED AS 1) "IVC" ERROR WITH VOLUME
11579                                   ;VALID ZERO, OR 2) ERRONEOUS "IVC" ERROR WITH VOLUME VALID SET.
11580                                   ;
11581                                   ;       ."SKI" IS REPORTED IF SET
11582                                   ;
11583                                   ;       ."DVC" IS REPORTED IF SET
11584                                   ;
11585                                   ;       .AN ERROR IS REPORTED IF "MOL"=0, OR "PIP"=1, OR "ATA"=0,
11586                                   ;OR "VV"=0
11587                                   ;
11588                                   ;CALL:
11589                                   ;(1)    JSR     PC,SCHSTS
11590                                   ;(2)    BR      ??              RETURN HERE IF NO ERROR
11591                                   ;(3)    NOP                     RETURN HERE TO REPORT ERROR
11592                                   ;(4)    ERROR                   SUBROUTINE WILL LOAD ERROR #
11593                                   ;(5)    JSR     PC,@(SP)+       GO BACK FOR MORE CHECKS
11594                                   ;(6)    ??                      RETURN AFTER ALL ERRORS REPORTED
11595
11596   057516                          SCHSTS:
11597
11598                                   ;CLEAR USER'S ERROR CALL
11599   057516  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11600   057522  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
11601   057526  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR BR
11602   057532  005037  061060                CLR     200$            ;CLEAR STATUS FLAGS
11603                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING REMOTE
11604                                   ;REGISTERS, I.E.  "PAR"=1 AND "DPE"=0.
11605   057536  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
11606   057544  001431                      BEQ     10$             ;NO!!
11607   057546  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS IT CONTROL BUS ERROR??
11608   057554  001025                      BNE     10$             ;PROBABLY NOT!!
11609
11610                                   ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL
11611   057556  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11612   057564  042737  000010  001140        BIC     #PAR,$GDDAT
```

```
11613  057572  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11614  057600  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11615  057604  112776  000050  000000          MOVB    #50,@(SP)               ;WRITE ERROR NUMBER IN CALL
11616  057612  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11617  057616  004736                          JSR     PC,@(SP)+               ;REPORT ERROR
11618  057620  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11619  057624  000240                          NOP
11620  057626  000430                          BR      15$                     ;SKIP FURTHER ERROR CHECKS
11621  057630                          10$:
11622
11623                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN READING REMOTE
11624                                   ;REGISTERS, I.E., "MCPE"=1.
11625  057630  032737  020000  001326          BIT     #MCPE,RMCS1I            ;WAS PARITY ERROR DETECTED??
11626  057636  001426                          BEQ     20$                     ;NO!!
11627
11628                                   ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL.
11629  057640  013737  001326  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
11630  057646  042737  020000  001140          BIC     #MCPE,$GDDAT
11631  057654  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
11632  057662  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11633  057666  112776  000013  000000          MOVB    #13,@(SP)               ;WRITE ERROR NUMBER
11634  057674  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11635  057700  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11636  057702  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11637  057706  000240                          NOP
11638  057710  000137  061032          15$:    JMP     150$                    ;OMIT STATUS CHECKING
11639
11640  057714                          20$:
11641
11642                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK, SECTOR AND CYLINDER
11643                                   ;ADDRESS USED DURING SEARCH.  ALSO SET "SKI" FLAG IF CYLINDER ADDRESS
11644                                   ;IS TOO LARGE
11645  057714  012737  002000  001140          MOV     #IAE,$GDDAT             ;SETUP FOR IAE=1
11646  057722  052737  040000  061060          BIS     #SKI,200$               ;SETUP FOR SKI=1
11647  057730  023727  001432  001466          CMP     RMDC0,#822.             ;CYLINDER > 822??
11648  057736  101025                          BHI     30$                     ;YES - CYLINDER IS INVALID
11649  057740  042737  040000  061060          BIC     #SKI,200$               ;"SKI" SHOULD BE ZERO
11650  057746  123727  001404  000037          CMPB    RMDA0,#31.              ;SECTOR > 31.??
11651  057754  101016                          BHI     30$                     ;YES - SECTOR IS INVALID
11652  057756  123727  001404  000035          CMPB    RMDA0,#29.              ;SECTOR > 29??
11653  057764  101404                          BLOS    25$                     ;NO!!
11654  057766  032737  010000  001360          BIT     #FMT16,RMOFI            ;18 BIT FORMAT??
11655  057774  001406                          BEQ     30$                     ;YES - SECTOR IS INVALID
11656  057776  123727  000007  000004  25$:    CMPB    RMDA+1,#4               ;TRACK > 4??
11657  060004  101002                          BHI     30$                     ;YES - INVALID TRACK
11658  060006  005037  001140                  CLR     $GDDAT                  ;IAE SHOULD BE ZERO
11659  060012                          30$:
11660
11661                                   ;COMPARE EXPECTED AND RECIVED "IAE" STATUS
11662  060012  013737  001342  001142          MOV     RMER1I,$BDDAT           ;GET RECEIVED IAE
11663  060020  042737  175777  001142          BIC     #^CIAE,$BDDAT
11664  060026  023737  001140  001142          CMP     $GDDAT,$BDDAT           ;IS IAE CORRECT??
11665  060034  001004                          BNE     40$                     ;NO!!
11666  060036  042737  040000  061060          BIC     #SKI,200$               ;SKI SHOULD BE ZERO
11667  060044  000413                          BR      50$
11668  060046                          40$:
```

```
11669
11670                                          ;REPORT INCORRECT IAE STATUS VIA USER'S ERROR CALL
11671  060046  062716  000004                         ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11672  060052  112776  000257  000000                 MOVB    #257,@(SP)              ;WRITE ERROR NUMBER IN CALL
11673  060060  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11674  060064  004736                                 JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11675  060066  162716  000010                         SUB     #10,(SP)                ;RETURN SP TO NO ERROR
11676  060072  000240                                 NOP
11677  060074                                  50$:
11678
11679                                          ;SEE IF "SKI" OR "DVC" OR "IVC" IS SET
11680  060074  032737  050200  001370                 BIT     #SKI!DVC!IVC,RMER2I     ;ARE ANY BITS SET??
11681  060102  001531                                 BEQ     90$                     ;NO!!
11682
11683                                          ;REPORT ERROR IF "SKI" IS SET AND "SKI" FLAG IS NOT SET
11684  060104  013737  001370  001142                 MOV     RMER2I,$BDDAT   ;RECEIVED "SKI" STATUS
11685  060112  042737  137777  001142                 BIC     #^CSKI,$BDDAT
11686  060120  013737  061060  001140                 MOV     200$,$GDDAT     ;EXPECTED "SKI" STATUS
11687  060126  042737  137777  001140                 BIC     #^CSKI,$GDDAT
11688  060134  023737  001140  001142                 CMP     $GDDAT,$BDDAT   ;IS "SKI" OK??
11689  060142  001422                                 BEQ     60$             ;YES-CHANGE ERROR NUMBER
11690  060144  062716  000004                         ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11691  060150  112776  000263  000000                 MOVB    #263,@(SP)              ;WRITE ERROR NUMBER
11692  060156  032737  040000  001140                 BIT     #SKI,$GDDAT     ;SHOUL "SKI" BE SET??
11693  060164  001403                                 BEQ     55$             ;NO!!
11694  060166  112776  000267  000000                 MOVB    #267,@(SP)      ;YES-CHANGE ERROR NUMBER
11695  060174  162716  000002                  55$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11696  060200  004736                                 JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11697  060202  162716  000010                         SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11698  060206  000240                                 NOP
11699  060210                                  60$:
11700
11701                                          ;REPORT "IVC" ERROR AS
11702                                          ;       .IVC DUE TO LOSS OF VOLUME VALID
11703                                          ;       .ERRONEOUS IVC WITH VOLUME VALID SET
11704  060210  032737  010000  001370                 BIT     #IVC,RMER2I                     ;IS IVC SET??
11705  060216  001433                                 BEQ     80$                     ;NO!!
11706  060220  013737  001370  001140                 MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11707  060226  042737  010000  001140                 BIC     #IVC,$GDDAT
11708  060234  013737  001370  001142                 MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11709  060242  062716  000004                         ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11710  060246  112776  000264  000000                 MOVB    #264,@(SP)              ;WRITE ERROR NUMBER IN CALL
11711  060254  032737  000100  001340                 BIT     #VV,RMDSI               ;WAS VOLUME VALID??
11712  060262  001403                                 BEQ     70$                     ;NO!!
11713  060264  112776  000265  000000                 MOVB    #265,@(SP)              ;YES - CHANGE ERROR NUMBER
11714  060272  162716  000002                  70$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11715  060276  004736                                 JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11716  060300  162716  000010                         SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11717  060304  000240                                 NOP
11718  060306                                  80$:
11719
11720                                          ;REPORT ANY DEVICE FAULT, I.E., "DVC"=1
11721  060306  032737  000200  001370                 BIT     #DVC,RMER2I                     ;WAS THERE A DEVICE FAULT??
11722  060314  001424                                 BEQ     90$                     ;NO!!
11723  060316  013737  001370  001140                 MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11724  060324  042737  000200  001140                 BIC     #DVC,$GDDAT
```

```
11725  060332  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11726  060340  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11727  060344  112776  000266  000000          MOVB    #266,@(SP)              ;WRITE ERROR NUMBER IN CALL
11728  060352  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11729  060356  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11730  060360  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11731  060364  000240                          NOP
11732  060366                          90$:
11733
11734                                  ;REPORT ANY "OPI" ERROR AS
11735                                  ;       ."OPI" BECAUSE MEDIUM IS NOT ONLINE
11736                                  ;       ."OPI" BECAUSE "ON CYLINDER" DIDN'T DROP
11737  060366  032737  020000  001342          BIT     #OPI,RMER1I             ;WAS OPI SET??
11738  060374  001433                          BEQ     110$                    ;NO!!
11739  060376  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11740  060404  042737  020000  001140          BIC     #OPI,$GDDAT
11741  060412  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11742  060420  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11743  060424  112776  000270  000000          MOVB    #270,@(SP)              ;SETUP ERROR FOR MOL=0
11744  060432  032737  010000  001340          BIT     #MOL,RMDSI              ;WAS MOL 0??
11745  060440  001403                          BEQ     105$                    ;YES!!
11746  060442  112776  000271  000000          MOVB    #271,@(SP)              ;MOL WAS 1 - CHANGE ERROR
11747  060450  162716  000002          105$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11748  060454  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11749  060456  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11750  060462  000240                          NOP
11751  060464                          110$:
11752
11753                                  ;SEE IF "ATA"="MOL"="VV"=1 AND "PIP"=0
11754  060464  013746  001340                  MOV     RMDSI,-(SP)             ;GET DRIVE STATUS
11755  060470  042716  047677                  BIC     #^C<ATA!PIP!MOL!VV>,(SP)
11756  060474  022726  110100                  CMP     #ATA!MOL!VV,(SP)+       ;IS DRIVE STATUS CORRECT??
11757  060500  001554                          BEQ     150$                    ;YES!!
11758
11759                                  ;REPORT AN ERROR IF MOL=0 AND OPI ERROR WAS NOT REPORTED, I.E., OPI=0
11760  060502  032737  010000  001340          BIT     #MOL,RMDSI              ;WAS MEDIUM OFF LINE??
11761  060510  001030                          BNE     120$                    ;NO!!
11762  060512  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11763  060520  052737  010000  001140          BIS     #MOL,$GDDAT
11764  060526  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11765  060534  032737  020000  001342          BIT     #OPI,RMER1I             ;WAS OPI REPORTED BEFORE??
11766  060542  001013                          BNE     120$                    ;YES - DON'T REPORT MOL=0
11767  060544  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11768  060550  112776  000272  000000          MOVB    #272,@(SP)              ;WRITE ERROR NUMBER IN CALL
11769  060556  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11770  060562  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11771  060564  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11772  060570  000240                          NOP
11773  060572                          120$:
11774
11775                                  ;REPORT AN ERROR IF PIP IS STIL SET AND SKI IS RESET
11776  060572  032737  020000  001340          BIT     #PIP,RMDSI              ;IS POSITIONING IN PROGRESS??
11777  060600  001430                          BEQ     130$                    ;NO!!
11778  060602  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11779  060610  001024                          BNE     130$            ;YES-DONT REPORT PIP
11780  060612  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
```

```
11781  060620  042737  020000  001140          BIC     #PIP,$GDDAT
11782  060626  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11783  060634  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11784  060640  112776  000273  000000          MOVB    #273,@(SP)              ;WRITE ERROR NUMBER IN CALL
11785  060646  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11786  060652  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11787  060654  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11788  060660  000240                          NOP
11789  060662                          130$:
11790
11791                                  ;REPORT AN ERROR IF VOLUME IS NOT VALID AND IVC=0
11792  060662  032737  000100  001340          BIT     #VV,RMDSI               ;IS VOLUME VALID??
11793  060670  001030                          BNE     140$                    ;YES!!
11794  060672  032737  010000  001370          BIT     #IVC,RMER2I             ;WAS IVC DETECTED??
11795  060700  001024                          BNE     140$                    ;YES - DON'T REPORT VV=0
11796  060702  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11797  060710  052737  000100  001140          BIS     #VV,$GDDAT
11798  060716  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11799  060724  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USERS ERROR CALL
11800  060730  112776  000350  000000          MOVB    #350,@(SP)              ;WRITE ERROR NUMBER IN CALL
11801  060736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11802  060742  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11803  060744  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11804  060750  000240                          NOP
11805  060752                          140$:
11806
11807                                  ;REPORT AN ERROR IF ATTENTION IS NOT SET
11808  060752  032737  100000  001340          BIT     #ATA,RMDSI              ;IS ATA ON??
11809  060760  001024                          BNE     150$                    ;YES!!
11810  060762  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11811  060770  052737  100000  001140          BIS     #ATA,$GDDAT
11812  060776  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11813  061004  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11814  061010  112776  000351  000000          MOVB    #351,@(SP)              ;WRITE ERROR NUMBER IN CALL
11815  061016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11816  061022  004736                          JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
11817  061024  162716  000010                  SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
11818  061030  000240                          NOP
11819  061032                          150$:
11820
11821                                  ;AUGMENT THE RETURN ADDRESS IF AN ERROR WAS DETECTED
11822  061032  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
11823  061036  105776  000000                  TSTB    @(SP)                   ;WAS ERROR FOUND??
11824  061042  001403                          BEQ     160$                    ;NO!!
11825  061044  062716  000004                  ADD     #4,(SP)                 ;YES - CHANGE RETURN
11826  061050  000402                          BR      170$
11827  061052  162716  000004          160$:   SUB     #4,(SP)                 ;NO ERROR FOUND
11828  061056  000207                  170$:   RTS     PC                      ;RETURN TO USER
11829
11830  061060  000000                  200$:   .WORD                           ;STORAGE FOR FLAGS
```

```
11831                                   .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
11832
11833                                   ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
11834                                   ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
11835                                   ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
11836                                   ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
11837
11838                                   ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
11839                                   ;IF TRUE:
11840
11841                                   ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
11842                                   ;THAT MOL IS ASSUMED TO HAVE BEEN SET
11843                                   ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
11844                                   ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
11845                                   ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
11846                                   ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
11847
11848                                   ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
11849
11850                                   ;(1)    JSR     PC,STCDRVSTS
11851                                   ;       BR      ???                     RETURN HERE IF NO ERROR
11852                                   ;       NOP                             RETURN HERE TO REPORT AN ERROR
11853                                   ;       ERROR                           ERROR NUMBER DEFINED BY SUB
11854                                   ;       JSR     PC,@(SP)+               GO BACK TO SUB FOR MORE ERROR CHECKS
11855                                   ;       ???                             RETURN HERE IF NO MORE ERRORS
11856
11857   061062                          STCDRVSTS:
11858
11859                                   ;CLEAR USER'S ERROR CALL
11860   061062  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11861   061066  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
11862   061072  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
11863                                   ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
11864   061076  013746  001340                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
11865   061102  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
11866   061106  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
11867   061112  001524                          BEQ     30$             ;YES!!
11868
11869                                   ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
11870   061114  032737  010000  001340          BIT     #MOL,RMDSI      ;IS MOL ON ??
11871   061122  001030                          BNE     10$             ;YES!!
11872   061124  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
11873   061132  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
11874   061134  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11875   061142  052737  010000  001140          BIS     #MOL,$GDDAT
11876   061150  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11877   061156  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11878   061162  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
11879   061170  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11880   061174  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11881   061176  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
11882  061202  000240                          NOP
11883  061204                          10$:
11884
11885                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11886  061204  032737  000100  001340          BIT     #VV,RMDSI       ;IS "VV" = 0??
11887  061212  001030                          BNE     20$             ;NO!!
11888  061214  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
11889  061222  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
11890  061224  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11891  061232  052737  000100  001340          BIS     #VV,RMDSI
11892  061240  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11893  061246  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11894  061252  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
11895  061260  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11896  061264  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11897  061266  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11898  061272  000240                          NOP
11899  061274                          20$:
11900
11901                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11902  061274  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11903  061302  001430                          BEQ     30$             ;NO!!
11904  061304  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11905  061312  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
11906  061314  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11907  061322  042737  020000  001140          BIC     #PIP,$GDDAT
11908  061330  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11909  061336  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11910  061342  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11911  061350  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11912  061354  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11913  061356  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11914  061362  000240                          NOP
11915  061364                          30$:
11916
11917                                  ;SEE IF "SKI" = "DVC" = 0
11918  061364  013746  001370                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
11919  061370  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
11920  061374  001460                          BEQ     60$             ;BRANCH IF NO ERROR
11921  061376                          40$:
11922
11923                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11924  061376  032737  000200  001370          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
11925  061404  001424                          BEQ     50$             ;NO!!
11926  061406  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11927  061414  042737  000200  001140          BIC     #DVC,$GDDAT
11928  061422  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11929  061430  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
11930  061434  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11931  061442  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11932  061446  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11933  061450  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11934  061454  000240                          NOP
11935  061456                          50$:
11936
11937                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
11938  061456  032737  040000  001370          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
11939  061464  001424                          BEQ     60$             ;NO!!
11940  061466  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11941  061474  042737  040000  001140          BIC     #SKI,$GDDAT
11942  061502  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11943  061510  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11944  061514  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
11945  061522  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11946  061526  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11947  061530  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11948  061534  000240                          NOP
11949  061536                          60$:
11950
11951                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11952  061536  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11953  061542  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11954  061546  001403                          BEQ     70$             ;NO!!
11955  061550  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
11956  061554  000402                          BR      80$
11957  061556  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
11958  061562  000240          80$:    NOP
11959  061564  000207                          RTS     PC              ;RETURN TO USER
```

```
11960                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
11961
11962                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
11963                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
11964                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
11965                                   ;THE MASKS ARE APPLIED.
11966
11967                                   ;CALL:
11968                                   ;(1)    JSR     PC,CMPERRSTS
11969                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
11970                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
11971                                   ;       BR      ???             RETURN HERE IF NO ERROR
11972                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11973                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11974                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11975                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11976
11977                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
11978                                   ;BE ZERO
11979
11980   061566                          CMPERRSTS:
11981
11982                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
11983   061566  013737  001342  001176          MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
11984   061574  047637  000000  001176          BIC     @(SP),$TMP1     ;MASK   RMER1
11985   061602  062716  000002                  ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
11986   061606  013737  001370  001200          MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
11987   061614  047637  000000  001200          BIC     @(SP),$TMP2     ;MASK RMER2
11988
11989
11990                                   ;CLEAR USER'S ERROR CALL
11991   061622  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
11992   061626  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
11993   061632  162716  000004                  SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
11994
11995                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
11996   061636  005737  001176                  TST     $TMP1           ;ANY ERRORS TO REPORT??
11997   061642  001420                          BEQ     5$              ;NO !!
11998   061644  013737  001176  001142          MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
11999   061652  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12000   061656  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12001   061662  112776  000066  000000          MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
12002   061670  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12003   061674  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12004   061676  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
12005   061702  000240                          NOP
12006   061704                          5$:
12007
12008
12009                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
12010   061704  005737  001200                  TST     $TMP2           ;ANY ERRORS IN RMER2?
12011   061710  001420                          BEQ     10$             ;NO!!
12012
12013   061712  013737  001200  001142          MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12014   061720  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12015   061724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
12016  061730  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
12017  061736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12018  061742  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12019  061744  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12020  061750  000240                          NOP
12021  061752                          10$:
12022
12023                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12024  061752  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12025  061756  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
12026  061762  001403                          BEQ     20$             ;NO!!
12027  061764  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12028  061770  000402                          BR      30$
12029  061772  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12030  061776  000207                  30$:    RTS     PC              ;RETURN TO USER
12031
12032
```

```
12033                                           .SBTTL  STOP AND SHUTDOWN SUBROUTINES
12034
12035   062000                                  STOP:
12036
12037
12038   062000  012746  000140                  ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
12039   062004  012746  062012                          MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
12040   062010  000002                                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12041   062012                                           RTI                     ;;POP NEW PC AND PS
12042   062012  000240                          64$:
12043                                                    NOP
12044   062014  012746  000300                  ;RAISE PRIORITY TO INHIBIT INTERRUPT
12045   062020  012746  062026                          MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
12046   062024  000002                                   MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
12047   062026                                           RTI                     ;;POP NEW PC AND PS
12048                                           65$:
12049   062026  000207                          10$:    RTS     PC              ;CONTINUE
12050
12051
12052   062030                                  SHUT:
12053   062030  104401  062052                          TYPE    ,100$           ;TYPE THE HALT MESSAGE
12054   062034  005737  000042                          TST     42              ;RUNNING STANDALONE ??
12055   062040  001402                                   BEQ     10$             ;YES !!
12056   062042  000137  042204                          JMP     $EOP            ;NO - GO TO END OF PASS
12057   062046                                  10$:
12058   062046  000137  005322                          JMP     START           ;GO TO START
12059   062052  042524  052123  053440  100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
12060   062060  051501  044040  046101
12061   062066  042524  006504  005012
12062   062074     000
12063           062076                          .EVEN
```

```
12064                                   .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
12065
12066                                   ;;*****************************************************************
12067                                   ;*SAVE R0-R5
12068                                   ;*CALL:
12069                                   ;*      SAVREG
12070                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
12071                                   ;*
12072                                   ;*TOP---(+16)
12073                                   ;* +2---(+18)
12074                                   ;* +4---R5
12075                                   ;* +6---R4
12076                                   ;* +8---R3
12077                                   ;*+10---R2
12078                                   ;*+12---R1
12079                                   ;*+14---R0
12080
12081  062076                           $SAVREG:
12082  062076  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12083  062100  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12084  062102  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12085  062104  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12086  062106  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12087  062110  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12088  062112  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
12089  062116  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
12090  062122  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
12091  062126  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
12092  062132  000002                          RTI
12093
12094                                   ;*RESTORE R0-R5
12095                                   ;*CALL:
12096                                   ;*      RESREG
12097  062134                           $RESREG:
12098  062134  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
12099  062140  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
12100  062144  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
12101  062150  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
12102  062154  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12103  062156  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12104  062160  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12105  062162  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12106  062164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12107  062166  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12108  062170  000002                          RTI
12109                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
12110
12111                                   ;;*****************************************************************
12112                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
12113                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
12114                                   ;*CALL:
12115                                   ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
12116                                   ;*      TYPBN                   ;;TYPE IT
12117
12118  062172  010146                   $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
12119  062174  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
12120  062200  000261                    SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
12121  062202  112737  000060  062244  1$:   MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
12122  062210  006101                    ROL     R1              ;;GET THIS BIT
12123  062212  001406                    BEQ     2$              ;;DONE?
12124  062214  105537  062244            ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
12125  062220  104401  062244            TYPE    ,$BIN           ;;GO TYPE THIS BIT
12126  062224  000241                    CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
12127  062226  000765                    BR      1$              ;;GO DO THE NEXT BIT
12128  062230  012601            2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
12129  062232  016666  000002  000004    MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12130  062240  012616                    MOV     (SP)+,(SP)
12131  062242  000002                    RTI                     ;;RETURN TO USER
12132  062244    000     000           $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
12133                                    .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
12134
12135                                    ;;*****************************************************************
12136                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
12137                                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
12138                                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
12139                                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
12140                                    ;*REPLACED WITH SPACES.
12141                                    ;*CALL:
12142                                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
12143                                    ;*      TYPDS                   ;;GO TO THE ROUTINE
12144
12145  062246                            $TYPDS:
12146  062246  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12147  062250  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12148  062252  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12149  062254  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12150  062256  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12151  062260  012746  020200                    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
12152  062264  016605  000020                    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
12153  062270  100004                            BPL     1$              ;;BR IF INPUT IS POS.
12154  062272  005405                            NEG     R5              ;;MAKE THE BINARY NUMBER POS.
12155  062274  112766  000055  000001            MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
12156  062302  005000                    1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
12157  062304  012703  062462                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
12158  062310  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
12159  062314  005002                    2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
12160  062316  016001  062452                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
12161  062322  160105                    3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
12162  062324  002402                            BLT     4$              ;;BR IF DONE
12163  062326  005202                            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
12164  062330  000774                            BR      3$
12165  062332  060105                    4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
12166  062334  005702                            TST     R2              ;;CHECK IF BCD DIGIT=0
12167  062336  001002                            BNE     5$              ;;FALL THROUGH IF 0
12168  062340  105716                            TSTB    (SP)            ;;STILL DOING LEADING 0'S?
12169  062342  100407                            BMI     7$              ;;BR IF YES
12170  062344  106316                    5$:     ASLB    (SP)            ;;MSD?
12171  062346  103003                            BCC     6$              ;;BR IF NO
12172  062350  116663  000001  177777            MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
12173  062356  052702  000060            6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
12174  062362  052702  000040            7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
12175  062366  110223                            MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
12176  062370  005720                        TST     (R0)+           ;;JUST INCREMENTING
12177  062372  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
12178  062376  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
12179  062400  003002                        BGT     8$              ;;GO TO EXIT
12180  062402  010502                        MOV     R5,R2           ;;GET THE LSD
12181  062404  000764                        BR      6$              ;;GO CHANGE TO ASCII
12182  062406  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
12183  062410  100003                        BPL     9$              ;;BR IF NO
12184  062412  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
12185  062420  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
12186  062422  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
12187  062424  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
12188  062426  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
12189  062430  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
12190  062432  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
12191  062434  104401  062462                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
12192  062440  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12193  062446  012616                        MOV     (SP)+,(SP)
12194  062450  000002                        RTI                     ;;RETURN TO USER
12195  062452  023420                $DTBL:  10000.
12196  062454  001750                        1000.
12197  062456  000144                        100.
12198  062460  000012                        10.
12199  062462  000004                $DBLK:  .BLKW   4
12200                                 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
12201
12202                                 ;;**************************************************************
12203                                 ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
12204                                 ;;*OCTAL (ASCII) NUMBER AND TYPE IT.
12205                                 ;;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
12206                                 ;;*CALL:
12207                                 ;;*     MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12208                                 ;;*     TYPOS                   ;;CALL FOR TYPEOUT
12209                                 ;;*     .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
12210                                 ;;*     .BYTE   M               ;;M=1 OR 0
12211                                 ;;*                                     ;;1=TYPE LEADING ZEROS
12212                                 ;;*                                     ;;0=SUPPRESS LEADING ZEROS
12213                                 ;;*
12214                                 ;;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
12215                                 ;;*$TYPOS OR $TYPOC
12216                                 ;;*CALL:
12217                                 ;;*     MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12218                                 ;;*     TYPON                   ;;CALL FOR TYPEOUT
12219                                 ;;*
12220                                 ;;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
12221                                 ;;*CALL:
12222                                 ;;*     MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12223                                 ;;*     TYPOC                   ;;CALL FOR TYPEOUT
12224
12225  062472  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
12226  062476  116637  000001  062715        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
12227  062504  112637  062717                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
12228  062510  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
12229  062514  000406                        BR      $TYPON
12230  062516  112737  000001  062715  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
12231  062524  112737  000006  062717        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
12232  062532  112737  000005  062714  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
12233  062540  010346                          MOV     R3,-(SP)        ;;SAVE R3
12234  062542  010446                          MOV     R4,-(SP)        ;;SAVE R4
12235  062544  010546                          MOV     R5,-(SP)        ;;SAVE R5
12236  062546  113704  062717                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
12237  062552  005404                          NEG     R4
12238  062554  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
12239  062560  110437  062716                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
12240  062564  113704  062715                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
12241  062570  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
12242  062574  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
12243  062576  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
12244  062600  000404                          BR      3$              ;;GO DO MSB
12245  062602  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
12246  062604  006105                          ROL     R5
12247  062606  006105                          ROL     R5
12248  062610  010503                          MOV     R5,R3
12249  062612  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
12250  062614  105337  062716                  DECB    $OMODE          ;;TYPE THIS DIGIT?
12251  062620  100016                          BPL     7$              ;;BR IF NO
12252  062622  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
12253  062626  001002                          BNE     4$              ;;TEST FOR 0
12254  062630  005704                          TST     R4              ;;SUPPRESS THIS 0?
12255  062632  001403                          BEQ     5$              ;;BR IF YES
12256  062634  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
12257  062636  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
12258  062642  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
12259  062646  110337  062712                  MOVB    R3,8$           ;;SAVE FOR TYPING
12260  062652  104401  062712                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
12261  062656  105337  062714          7$:     DECB    $OCNT           ;;COUNT BY 1
12262  062662  003347                          BGT     2$              ;;BR IF MORE TO DO
12263  062664  002402                          BLT     6$              ;;BR IF DONE
12264  062666  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
12265  062670  000744                          BR      2$              ;;GO DO THE LAST DIGIT
12266  062672  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
12267  062674  012604                          MOV     (SP)+,R4        ;;RESTORE R4
12268  062676  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12269  062700  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
12270  062706  012616                          MOV     (SP)+,(SP)
12271  062710  000002                          RTI                     ;;RETURN
12272  062712     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
12273  062713     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
12274  062714     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
12275  062715     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
12276  062716  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
12277                          .SBTTL  TYPE ROUTINE
12278
12279                          ;;*********************************************************************
12280                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
12281                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
12282                          ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
12283                          ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
12284                          ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
12285                          ;*
12286                          ;*CALL:
12287                          ;*1) USING A TRAP INSTRUCTION
```

```
12288                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
12289                                   ;*OR
12290                                   ;*      TYPE
12291                                   ;*      MESADR
12292                                   ;*
12293
12294  062720  105737  001173           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
12295  062724  100002                           BPL     1$              ;;BR IF YES
12296  062726  000000                           HALT                    ;;HALT HERE IF NO TERMINAL
12297  062730  000430                           BR      3$              ;;LEAVE
12298  062732  010046                   1$:     MOV     R0,-(SP)        ;;SAVE R0
12299  062734  017600  000002                   MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
12300  062740  122737  000001  001242           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
12301  062746  001011                           BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
12302  062750  132737  000100  001243           BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
12303  062756  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
12304  062760  010037  062770                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
12305  062764  004737  065736                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
12306  062770  000000                   61$:    .WORD   0               ;;MESSAGE ADDRESS
12307  062772  132737  000040  001243   62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
12308  063000  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
12309  063002  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
12310  063004  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
12311  063006  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
12312  063010  012600                   60$:    MOV     (SP)+,R0        ;;RESTORE R0
12313  063012  062716  000002           3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
12314  063016  000002                           RTI                     ;;RETURN
12315  063020  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
12316  063024  001430                           BEQ     8$
12317  063026  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
12318  063032  001006                           BNE     5$
12319  063034  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
12320  063036  104401                           TYPE                    ;;TYPE A CR AND LF
12321  063040  001217                           $CRLF
12322  063042  105037  063176                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
12323  063046  000755                           BR      2$              ;;GET NEXT CHARACTER
12324  063050  004737  063132           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
12325  063054  123726  001172           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
12326  063060  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
12327  063062  013746  001170                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
12328                                           ;;AND THE NULL CHAR.
12329  063066  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
12330  063072  002770                           BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
12331  063074  004737  063132                   JSR     PC,$TYPEC       ;;GO TYPE A NULL
12332  063100  105337  063176                   DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
12333  063104  000770                           BR      7$              ;;LOOP
12334
12335                                   ;HORIZONTAL TAB PROCESSOR
12336
12337  063106  112716  000040           8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
12338  063112  004737  063132           9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
12339  063116  132737  000007  063176           BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
12340  063124  001372                           BNE     9$              ;;TAB STOP
12341  063126  005726                           TST     (SP)+           ;;POP SPACE OFF STACK
12342  063130  000724                           BR      2$              ;;GET NEXT CHARACTER
12343  063132  105777  116026           $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
12344  063136  100375                        BPL     $TYPEC
12345  063140  116677  000002  116020        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
12346  063146  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
12347  063154  001003                        BNE     1$              ;;BRANCH IF NO
12348  063156  105037  063176                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
12349  063162  000406                        BR      $TYPEX          ;;EXIT
12350  063164  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
12351  063172  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
12352  063174  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
12353  063176  000000              $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
12354  063200  000207              $TYPEX: RTS     PC
12355
12356                                        .SBTTL  SCOPE HANDLER ROUTINE
12357
12358                                        ;;**********************************************************************
12359                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
12360                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
12361                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
12362                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
12363                                        ;*SW14=1        LOOP ON TEST
12364                                        ;*SW11=1        INHIBIT ITERATIONS
12365                                        ;*SW09=1        LOOP ON ERROR
12366                                        ;*SW08=1        LOOP ON TEST IN SWR<7:0>
12367                                        ;*CALL
12368                                        ;*      SCOPE           ;;SCOPE=IOT
12369
12370  063202                        $SCOPE:
12371  063202  104410                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
12372  063204  004737  062000                JSR     PC,STOP
12373  063210  032777  040000  115736  1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
12374  063216  001131                        BNE     $OVER           ;;YES IF SW14=1
12375                                        ;*****START OF CODE FOR THE XOR TESTER*****
12376  063220  000416              $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
12377                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
12378  063222  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
12379  063226  012737  063246  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
12380  063234  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
12381  063240  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
12382  063244  000500                        BR      $SVLAD          ;;GO TO THE NEXT TEST
12383  063246  022626              5$:       CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
12384  063250  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
12385  063254  000440                        BR      7$              ;;LOOP ON THE PRESENT TEST
12386  063256                        6$:;*****END OF CODE FOR THE XOR TESTER*****
12387  063256  032777  000400  115670        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
12388  063264  001421                        BEQ     2$              ;;BR IF NO
12389  063266  005046                        CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
12390  063270  117716  115660                MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
12391  063274  001414                        BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
12392  063276  022716  000072                CMP     #72,(SP)        ;;CHECK THE NUMBER IN THE SWR
12393  063302  002411                        BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
12394  063304  011637  001116                MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
12395  063310  005316                        DEC     (SP)            ;;BACKUP BY ONE
12396  063312  006316                        ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
12397  063314  062716  063520                ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
12398  063320  013637  001122                MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
12399  063324  000466                        BR      $OVER           ;;GO LOOP ON THE TEST
```

```
12400  063326  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
12401  063330  105737  001117            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
12402  063334  001421                            BEQ     3$              ;;BR IF NO
12403  063336  123737  001131  001117            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
12404  063344  101015                            BHI     3$              ;;BR IF NO
12405  063346  032777  001000  115600            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
12406  063354  001404                            BEQ     4$              ;;BR IF NO
12407  063356  013737  001124  001122  7$:       MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
12408  063364  000446                            BR      $OVER
12409  063366  105037  001117            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
12410  063372  005037  001206                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
12411  063376  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
12412  063400  032777  004000  115546  3$:       BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
12413  063406  001011                            BNE     1$              ;;BR IF YES
12414  063410  005737  001230                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
12415  063414  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
12416  063416  005237  001120                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
12417  063422  023737  001206  001120            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
12418  063430  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
12419  063432  012737  000001  001120  1$:       MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
12420  063440  013737  063516  001206            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
12421  063446  105237  001116            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
12422  063452  113737  001116  001226            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
12423  063460  011637  001122                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
12424  063464  011637  001124                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
12425  063470  005037  001210                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
12426  063474  112737  000001  001131            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
12427  063502  013777  001116  115446  $OVER:    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
12428  063510  013716  001122                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
12429  063514  000002                            RTI                     ;;FIXES PS
12430  063516  000012            $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
12431  063520                    $SWO8TBL:
12432  063520  007142                            .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
12433  063522  007340                            .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
12434  063524  007524                            .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
12435  063526  007706                            .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
12436  063530  010572                            .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
12437  063532  011130                            .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
12438  063534  011334                            .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
12439  063536  011660                            .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
12440  063540  012040                            .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
12441  063542  013112                            .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
12442  063544  013412                            .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
12443  063546  013644                            .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
12444  063550  014166                            .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
12445  063552  014512                            .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
12446  063554  014770                            .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
12447  063556  015262                            .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
12448  063560  015532                            .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
12449  063562  015762                            .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
12450  063564  016306                            .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
12451  063566  016726                            .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
12452  063570  017272                            .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
12453  063572  017664                            .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
12454  063574  020340                            .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
12455  063576  020672                            .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
```

```
12456   063600  021214                  .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
12457   063602  021576                  .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
12458   063604  022114                  .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
12459   063606  022700                  .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
12460   063610  023424                  .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
12461   063612  024214                  .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
12462   063614  025000                  .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
12463   063616  025624                  .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
12464   063620  026162                  .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
12465   063622  026622                  .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
12466   063624  027046                  .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
12467   063626  027404                  .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
12468   063630  027706                  .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
12469   063632  030210                  .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
12470   063634  030536                  .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
12471   063636  031060                  .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
12472   063640  031406                  .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
12473   063642  031734                  .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
12474   063644  032262                  .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
12475   063646  032650                  .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
12476   063650  033264                  .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
12477   063652  033676                  .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
12478   063654  034236                  .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
12479   063656  034600                  .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
12480   063660  035146                  .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
12481   063662  035620                  .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
12482   063664  036276                  .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
12483   063666  037004                  .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
12484   063670  037364                  .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
12485   063672  037752                  .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
12486   063674  040356                  .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
12487   063676  040736                  .WORD   TST70+2         ;;STARTING ADDRESS OF TEST 70
12488   063700  041302                  .WORD   TST71+2         ;;STARTING ADDRESS OF TEST 71
12489   063702  041650                  .WORD   TST72+2         ;;STARTING ADDRESS OF TEST 72
12490                           .SBTTL  ERROR HANDLER ROUTINE
12491
12492                           ;;*******************************************************************
12493                           ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
12494                           ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
12495                           ;*AND GO TO ERRTYP ON ERROR
12496                           ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
12497                           ;*SW15=1         HALT ON ERROR
12498                           ;*SW13=1         INHIBIT ERROR TYPEOUTS
12499                           ;*SW10=1         BELL ON ERROR
12500                           ;*SW09=1         LOOP ON ERROR
12501                           ;*CALL
12502                           ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
12503
12504   063704                  $ERROR:
12505   063704  104410                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12506   063706  105237  001117  7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
12507   063712  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
12508   063714  013777  001116  115234  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
12509   063722  032777  002000  115224  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
12510   063730  001402                  BEQ     1$              ;;NO - SKIP
12511   063732  104401  001212          TYPE    ,$BELL          ;;RING BELL
```

```
12512  063736  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
12513  063742  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
12514  063746  162737  000002  001132          SUB     #2,$ERRPC
12515  063754  117737  115152  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
12516  063762  032777  020000  115164          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
12517  063770  001004                          BNE     20$             ;;SKIP TYPEOUTS
12518  063772  004737  042414                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
12519  063776  104401  001217                  TYPE    ,$CRLF
12520  064002                          20$:
12521  064002  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
12522  064010  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
12523  064012  113737  001130  064024          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
12524  064020  004737  065746                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
12525  064024     000                  21$:    .BYTE   0
12526  064025     000                          .BYTE   0
12527  064026  000777                  22$:    BR      22$             ;;APT ERROR LOOP
12528  064030  005777  115120          2$:     TST     @SWR            ;;HALT ON ERROR
12529  064034  100002                          BPL     3$              ;;SKIP IF CONTINUE
12530  064036  000000                          HALT                    ;;HALT ON ERROR!
12531  064040  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12532  064042  032777  001000  115104  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
12533  064050  001402                          BEQ     4$              ;;BR IF NO
12534  064052  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
12535  064056  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
12536  064062  001402                          BEQ     5$              ;;BR IF NONE
12537  064064  013716  001210                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
12538  064070                          5$:
12539  064070  022737  042374  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
12540  064076  001001                          BNE     6$              ;;BRANCH IF NO
12541  064100  000000                          HALT                    ;;YES
12542  064102                          6$:
12543  064102  000002                          RTI                     ;;RETURN
12544                                  .SBTTL  TTY INPUT ROUTINE
12545
12546                                  ;;*******************************************************************
12547                                  .ENABL  LSB
12548  064104  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
12549  064106  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
12550  064110  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
12551  064112  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
12552          064113                  $TKQEND=.
12553          064114                  .EVEN
12554
12555                                  ;*TK INITIALIZE ROUTINE
12556                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12557                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12558                                  ;
12559                                  ;*CALL:
12560                                  ;*      JSR     PC,$TKINT
12561                                  ;*      RETURN
12562                                  ;
12563  064114  005037  064104          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
12564  064120  012737  064112  064106          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
12565  064126  013737  064106  064110          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12566  064134  012737  064164  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
12567  064142  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
```

```
12568  064150  005777  115006          TST     @$TKB           ;;CLEAR DONE FLAG
12569  064154  012777  000100  114776  MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
12570  064162  000207                  RTS     PC              ;;RETURN TO CALLER
12571
12572                          ;*TK SERVICE ROUTINE
12573                          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12574                          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
12575                          ;*IT IN THE QUEUE.
12576                          ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
12577                          ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT)
12578                          ;
12579  064164  117746  114772  $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
12580  064170  042716  177600          BIC     #↑C177,(SP)     ;;STRIP THE JUNK
12581  064174  021627  000003          CMP     (SP),#3         ;;IS IT A CONTROL C?
12582  064200  001007                  BNE     1$              ;;BRANCH IF NO
12583  064202  104401  065300          TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
12584  064206  004737  064114          JSR     PC,$TKINT       ;;INIT THE KEYBOARD
12585  064212  005726                  TST     (SP)+           ;;CLEAN UP STACK
12586  064214  000137  062030          JMP     SHUT            ;;CONTROL C RESTART
12587  064220  021627  000007  1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
12588  064224  001004                  BNE     2$              ;;BRANCH IF NO
12589  064226  022737  000176  001154  CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
12590  064234  001500                  BEQ     6$              ;;GO TO SWR CHANGE
12591
12592  064236          2$:
12593  064236  022737  000001  064104  CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
12594  064244  001004                  BNE     3$              ;;BRANCH IF NO
12595  064246  104401  001212          TYPE    ,$BELL          ;;RING THE TTY BELL
12596  064252  005726                  TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
12597  064254  000451                  BR      5$              ;;EXIT
12598  064256  021627  000023  3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
12599  064262  001021                  BNE     32$             ;;BRANCH IF NO
12600  064264  005077  114670          CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12601  064270  005726                  TST     (SP)+           ;;CLEAN CHAR OFF STACK
12602  064272  105777  114662  31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
12603  064276  100375                  BPL     31$             ;;LOOP UNTIL ITS THERE
12604  064300  117746  114656          MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
12605  064304  042716  177600          BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
12606  064310  022627  000021          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
12607  064314  001366                  BNE     31$             ;;BRANCH IF NO
12608  064316  012777  000100  114634  MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
12609  064324  000002                  RTI                     ;;RETURN
12610  064326  005237  064104  32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
12611  064332  021627  000140          CMP     (SP),#140       ;;IS IT UPPER CASE?
12612  064336  002405                  BLT     4$              ;;BRANCH IF YES
12613  064340  021627  000175          CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
12614  064344  003002                  BGT     4$              ;;BRANCH IF YES
12615  064346  042716  000040          BIC     #40,(SP)        ;;MAKE IT UPPER CASE
12616  064352  112677  177530  4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
12617  064356  005237  064106          INC     $TKQIN          ;;UPDATE THE POINTER
12618  064362  023727  064106  064113  CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
12619  064370  001003                  BNE     5$              ;;BRANCH IF NO
12620  064372  012737  064112  064106  MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
12621  064400  000002          5$:     RTI                     ;;RETURN
12622
12623                          ;;*****************************************************************
```

```
12624                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12625                                   ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
12626                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12627                                   ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12628  064402  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
12629  064410  001124                          BNE     15$             ;;EXIT IF NOT
12630  064412  105777  114542                  TSTB    @$TKS           ;;IS A CHAR WAITING?
12631  064416  100121                          BPL     15$             ;;IF NOT, EXIT
12632  064420  117746  114536                  MOVB    @$TKB,-(SP)     ;;YES
12633  064424  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
12634  064430  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
12635  064434  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
12636                                                                  ;;AND EXIT
12637
12638                                   ;;*****************************************************************
12639                                   ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12640                                   ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12641                                   ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12642  064436  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
12643  064444  001674                          BEQ     2$              ;;BRANCH IF YES
12644  064446  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
12645  064450  004737  064114                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
12646  064454  005077  114500                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12647  064460  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
12648
12649  064466  104401  065312                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
12650  064472  104401  065317  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
12651  064476  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
12652  064502  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
12653  064504  104401  065330                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
12654  064510  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
12655  064512  005046                          CLR     -(SP)           ;;THE NEW SWR
12656  064514  105777  114440          7$:     TSTB    @$TKS           ;;CHAR THERE?
12657  064520  100375                          BPL     7$              ;;IF NOT TRY AGAIN
12658
12659  064522  117746  114434                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
12660  064526  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12661
12662  064532  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
12663  064536  001015                          BNE     9$              ;;BRANCH IF NOT
12664  064540  104401  065300                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
12665  064544  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
12666  064550  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
12667  064556  001003                          BNE     8$              ;;BRANCH IF NO
12668  064560  012777  000100  114372          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
12669  064566  000137  062030          8$:     JMP     SHUT            ;;CONTROL-C RESTART
12670
12671
12672  064572  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
12673  064576  001005                          BNE     10$             ;;BRANCH IF NOT
12674  064600  104401  065305                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
12675  064604  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
12676  064610  000737                          BR      19$             ;;LET'S TRY IT AGAIN
12677
12678
12679  064612  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
```

```
12660  064616  001022                        BNE     16$             ;;BRANCH IF NO
12661  064620  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
12662  064624  001403                        BEQ     11$             ;;BRANCH IF YES
12663  064626  016677  000002  114320        MOV     2(SP),@SWR      ;;SAVE NEW SWR
12664  064634  062706  000006          11$:  ADD     #6,SP           ;;CLEAR UP STACK
12665  064640  104401  001217          14$:  TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
12666  064644  123727  001151  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
12667  064652  001003                        BNE     15$             ;;BRANCH IF NOT
12668  064654  012777  000100  114276        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
12669  064662  000002          15$:  RTI                             ;;RETURN
12690  064664  004737  063132          16$:  JSR     PC,$TYPEC       ;;ECHO CHAR
12691  064670  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
12692  064674  002420                        BLT     18$             ;;BRANCH IF YES
12693  064676  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
12694  064702  003015                        BGT     18$             ;;BRANCH IF YES
12695  064704  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
12696  064710  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
12697  064714  001403                        BEQ     17$             ;;BRANCH IF YES
12698  064716  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
12699  064720  006316                        ASL     (SP)            ;;    CHAR OVER TO MAKE
12700  064722  006316                        ASL     (SP)            ;;    ROOM FOR NEW ONE.
12701  064724  005266  000002          17$:  INC     2(SP)           ;;KEEP COUNT OF CHAR
12702  064730  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
12703  064734  000667                        BR      7$              ;;GET THE NEXT ONE
12704  064736  104401  001216          18$:  TYPE    ,$QUES          ;;TYPE ?<CR><LF>
12705  064742  000720                        BR      20$             ;;SIMULATE CONTROL-U
12706                                 .DSABL  LSB
12707
12708
12709                                 ;;*****************************************************************
12710                                 ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
12711                                 ;*CALL:
12712                                 ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
12713                                 ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
12714                                 ;*                              ;;WITH PARITY BIT STRIPPED OFF
12715                                 ;
12716
12717  064744  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
12718  064746  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE PS
12719  064754  005066  000004                CLR     4(SP)           ;;GET READY FOR A CHARACTER
12720  064760  005046                        CLR     -(SP)           ;;PUT NEW PS ON STACK
12721  064762  012746  064770                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12722  064766  000002                        RTI                     ;;POP NEW PC AND PS
12723  064770                        64$:
12724  064770  005737  064104          1$:   TST     $TKCNT          ;;WAIT ON A CHARACTER
12725  064774  001775                        BEQ     1$
12726  064776  005337  064104                DEC     $TKCNT          ;;DECREMENT THE COUNTER
12727  065002  117766  177102  000004        MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
12728  065010  005237  064110                INC     $TKQOUT         ;;UPDATE THE POINTER
12729  065014  023727  064110  064113        CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
12730  065022  001003                        BNE     2$              ;;BRANCH IF NO
12731  065024  012737  064112  064110        MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
12732  065032  000002          2$:   RTI                             ;;RETURN
12733                                 ;;*****************************************************************
12734                                 ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
12735                                 ;*CALL:
```

```
12736                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
12737                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
12738                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
12739
12740  065034  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
12741  065036  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
12742  065040  012703  065270          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
12743  065044  022703  065300          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
12744  065050  101456                          BLOS    4$              ;;BR IF YES
12745  065052  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
12746  065054  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
12747  065056  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
12748  065062  001022                          BNE     5$              ;;BR IF NO
12749  065064  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
12750  065066  001007                          BNE     6$              ;;BR IF NO
12751  065070  112737  000134  065266          MOVB    #'\,9$          ;;TYPE A BACK SLASH
12752  065076  104401  065266                  TYPE    9$
12753  065102  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
12754  065106  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
12755  065110  020327  065270                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
12756  065114  103434                          BLO     4$              ;;BR IF YES
12757  065116  111337  065266                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
12758  065122  104401  065266                  TYPE    9$              ;;GO TYPE
12759  065126  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
12760  065130  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
12761  065132  001406                          BEQ     7$              ;;BR IF NO
12762  065134  112737  000134  065266          MOVB    #'\,9$          ;;TYPE A BACK SLASH
12763  065142  104401  065266                  TYPE    9$
12764  065146  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
12765  065150  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
12766  065154  001003                          BNE     8$              ;;BR IF NO
12767  065156  104401  065305                  TYPE    $CNTLU          ;;TYPE A CONTROL "U"
12768  065162  000726                          BR      1$              ;;GO START OVER
12769  065164  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
12770  065170  001011                          BNE     3$              ;;BRANCH IF NO
12771  065172  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
12772  065174  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
12773  065200  104401  065270                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
12774  065204  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
12775  065206  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
12776  065212  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
12777  065214  111337  065266          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
12778  065220  104401  065266                  TYPE    9$
12779  065224  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
12780  065230  001305                          BNE     2$              ;;LOOP IF NOT RETURN
12781  065232  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
12782  065236  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
12783  065242  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
12784  065244  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12785  065246  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
12786  065250  016666  000004  000002          MOV     4(SP),2(SP)     ;;       FIRST ASCII CHARACTER ON IT
12787  065256  012766  065270  000004          MOV     #$TTYIN,4(SP)
12788  065264  000002                          RTI                     ;;RETURN
12789  065266     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
12790  065267     000                          .BYTE   0               ;;TERMINATOR
12791  065270  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
```

```
12792  065300  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>     ;;CONTROL "C"
12793  065305     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
12794  065312  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
12795  065317     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
12796  065324  036440  000040
12797  065330  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
12798  065336  020075     000
12799          065342                  .EVEN
12800                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
12801
12802                                  ;;*****************************************************************
12803                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
12804                                  ;*CHANGE IT TO BINARY.
12805                                  ;*CALL:
12806                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
12807                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
12808                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
12809
12810  065342  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
12811  065344  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
12812  065352  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12813  065354  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12814  065356  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12815  065360  104412          1$:     RDLIN                   ;;READ AN ASCIZ LINE
12816  065362  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
12817  065364  005001                  CLR     R1              ;;CLEAR DATA WORD
12818  065366  005002                  CLR     R2
12819  065370  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
12820  065372  001412                  BEQ     3$              ;;IF ZERO GET OUT
12821  065374  006301                  ASL     R1              ;;*2
12822  065376  006102                  ROL     R2
12823  065400  006301                  ASL     R1              ;;*4
12824  065402  006102                  ROL     R2
12825  065404  006301                  ASL     R1              ;;*8
12826  065406  006102                  ROL     R2
12827  065410  042716  177770         BIC     #↑C7,(SP)       ;;STRIP THE ASCII JUNK
12828  065414  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
12829  065416  000764                  BR      2$              ;;LOOP
12830  065420  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
12831  065422  010166  000012         MOV     R1,12(SP)       ;;SAVE THE RESULT
12832  065426  010237  065442         MOV     R2,$HIOCT
12833  065432  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
12834  065434  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
12835  065436  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
12836  065440  000002                  RTI                     ;;RETURN
12837  065442  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
12838                                  .SBTTL  TRAP DECODER
12839
12840                                  ;;*****************************************************************
12841                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12842                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12843                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12844                                  ;*GO TO THAT ROUTINE.
12845
12846  065444  016646  000002  $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
12847  065450  042716  000020         BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
```

```
12848  065454  012746  065462          MOV     #1$,-(SP)       ;; T-BIT TRAPS
12849  065460  000002                  RTI                     ;;SET THE NEW STATUS
12850  065462  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
12851  065464  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
12852  065470  005740                  TST     -(R0)           ;;BACKUP BY 2
12853  065472  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
12854  065474  006300                  ASL     R0              ;;POSITION FOR INDEXING
12855  065476  016000  065516          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
12856  065502  000200                  RTS     R0              ;;GO TO ROUTINE
12857
12858
12859                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12860
12861  065504  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
12862  065506  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
12863  065514  000002                  RTI                     ;;RESTORE THE PSW
12864
12865                                  .SBTTL  TRAP TABLE
12866
12867                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12868                                  ;*BY THE "TRAP" INSTRUCTION.
12869
12870                                  ;       ROUTINE
12871                                  ;       -------
12872  065516  065504          $TRPAD: .WORD   $TRAP2
12873  065520  062720                  $TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
12874  065522  062516                  $TYPOC  ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12875  065524  062472                  $TYPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12876  065526  062532                  $TYPON  ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
12877  065530  062246                  $TYPDS  ;;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12878  065532  062172                  $TYPBN  ;;CALL=TYPBN     TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
12879
12880  065534  064472                  $GTSWR  ;;CALL=GTSWR     TRAP+7(104407)  GET SOFT-SWR SETTING
12881
12882  065536  064402                  $CKSWR  ;;CALL=CKSWR     TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
12883  065540  064744                  $RDCHR  ;;CALL=RDCHR     TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
12884  065542  065034                  $RDLIN  ;;CALL=RDLIN     TRAP+12(104412) TTY TYPEIN STRING ROUTINE
12885  065544  065342                  $RDOCT  ;;CALL=RDOCT     TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
12886  065546  062076                  $SAVREG ;;CALL=SAVREG    TRAP+14(104414) SAVE R0-R5 ROUTINE
12887  065550  062134                  $RESREG ;;CALL=RESREG    TRAP+15(104415) RESTORE R0-R5 ROUTINE
12888                                  .SBTTL  POWER DOWN AND UP ROUTINES
12889
12890                                  ;;*****************************************************************
12891                                  ;POWER DOWN ROUTINE
12892  065552  012737  065712  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12893  065560  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12894  065566  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12895  065570  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12896  065572  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12897  065574  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12898  065576  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12899  065600  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12900  065602  017746  113346          MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
12901  065606  010637  065716          MOV     SP,$SAVR6       ;;SAVE SP
12902  065612  012737  065624  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12903  065620  000000                  HALT
```

```
12904  065622  000776                          BR      .-2             ;;HANG UP
12905
12906                                  ;;*****************************************************************
12907                                  ;POWER UP ROUTINE
12908  065624  012737  065712  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12909  065632  013706  065716          MOV     $SAVR6,SP       ;;GET SP
12910  065636  005037  065716          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
12911  065642  005237  065716  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
12912  065646  001375                  BNE     1$              ;;OF  WORD
12913  065650  012677  113300          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
12914  065654  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
12915  065656  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
12916  065660  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
12917  065662  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
12918  065664  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
12919  065666  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
12920  065670  012737  065552  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12921  065676  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12922  065704  104401                          TYPE                    ;REPORT THE POWER FAILURE
12923  065706  065720          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
12924  065710  000002                  RTI
12925  065712  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12926  065714  000776                  BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
12927  065716  000000          $SAVR6: 0                       ;;PUT THE SP HERE
12928  065720  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
12929  065726  000122
12930                                  .EVEN
12931                                  .SBTTL  APT COMMUNICATIONS ROUTINE
12932
12933                                  ;;*****************************************************************
12934  065730  112737  000001  066174  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
12935  065736  112737  000001  066172  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
12936  065744  000403                  BR      $ATYC
12937  065746  112737  000001  066174  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
12938  065754                          $ATYC:
12939  065754  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12940  065756  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12941  065760  105737  066172          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
12942  065764  001450                  BEQ     5$              ;;IF NOT:  BR
12943  065766  122737  000001  001242  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
12944  065774  001031                  BNE     3$              ;;IF NOT:  BR
12945  065776  132737  000100  001243  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
12946  066004  001425                  BEQ     3$              ;;IF NOT:  BR
12947  066006  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
12948  066012  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12949  066020  005737  001222  1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
12950  066024  001375                  BNE     1$              ;;IF NOT:  WAIT
12951  066026  010037  001236          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
12952  066032  105720          2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
12953  066034  001376                  BNE     2$
12954  066036  163700  001236          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
12955  066042  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
12956  066044  010037  001240          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
12957  066050  012737  000004  001222  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
12958  066056  000413                  BR      5$
12959  066060  017637  000004  066104  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
```

```
12960  066066  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
12961  066074  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
12962  066100  004737  062720                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
12963  066104  000000                  4$:     .WORD   0
12964  066106                          5$:
12965  066106  105737  066174          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
12966  066112  001416                          BEQ     12$             ;;IF NOT:  BR
12967  066114  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
12968  066120  001413                          BEQ     12$             ;;IF NOT:  BR
12969  066122  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
12970  066126  001375                          BNE     11$             ;;IF NOT:  WAIT
12971  066130  017637  000004  001224          MOV     @4(SP),$FATAL   ;;GET ERROR #
12972  066136  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12973  066144  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
12974  066150  105037  066174          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
12975  066154  105037  066173                  CLRB    $LFLG           ;;CLEAR LOG FLAG
12976  066160  105037  066172                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
12977  066164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12978  066166  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12979  066170  000207                          RTS     PC              ;;RETURN
12980  066172     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
12981  066173     000          $LFLG:  .BYTE   0               ;;LOG FLAG
12982  066174     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
12983          066176                  .EVEN
12984          000200          APTSIZE=200
12985          000001          APTENV=001
12986          000100          APTSPOOL=100
12987          000040          APTCSUP=040
12988
12989                          .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

066176                          SCTMSG:
066176  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
066260  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
066306     051                  CLSPRN: .ASCII  @)@
066307     075     000          EQUALS: .ASCIZ  @=@
066311     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
066315     077     000          QSTMRK: .ASCIZ  @?@
066317                          HELPQST:
066317     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
066353                          UBUSQST:
066353     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@
066452  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
066511     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
066536  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
066565     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >1600000@
066617     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
066647     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
066673     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
066723     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
066757     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
067004                          CNSL07:
067004  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
067062  047040  046525  042502  .ASCII  @ NUMBER(S)@
067074  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
067143     015  041412  040510  XDPMG:  .ASCII  <CR><LF>/CHANGE XXDP PACK,CLEAR LOC 40/
067202  005015  042522  052123          .ASCIZ  <CR><LF>/RESTART THE PROGRAM/
067230  005015  047516  020124  NOTEX:  .ASCIZ  <CR><LF>/NOT EXIST DRIVE  /

                                .EVEN
```

```
                        .SBTTL  FUNCTION CODE TABLE

                        ;THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
                        ;EACH FUNCTION CODE.  BIT USAGE IS AS FOLLOWS:

                        ;       ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF ATA SHOULD BE SET WHEN THE FUNCTION CODE IS EXECUTED, OTHERWISE,
                        ;BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
                        ;NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
                        ;IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

                        ;       WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

                        ;       OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;THE WRITE ERRORS WHICH ARE ENABLED ARE "WLE", "WCF", "DPE", "UPE".

                        ;       IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

                        ;       AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
                        ;COMMAND.  THE ERRORS ENABLED BY THIS BIT ARE "TRE", "DLT", "NEM",
                        ;"MXF", "LBT", AND "AOE".

                        ;               BIT 08 IS NOT USED.

                        ;       HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;HEADER ERRORS INCLUDE "HCRC", "HCE", "FER", AND "BSE".

                        ;       ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
                        ;COMMAND.  THESE ERRORS INCLUDE "MDPE", "DCK", AND "ECH".

                        ;               BIT 05 IS NOT USED.

                        ;               BIT 04 IS NOT USED.

                        ;               BIT 03 IS NOT USED.

                        ;               BIT 02 IS NOT USED.

                        ;               BIT 01 IS NOT USED.

                        ;       ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

067254                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
067254  020000          .WORD   OPI                                     ;NOP
067256  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
067260  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
067262  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
067264  020000          .WORD   OPI                                     ;DRIVE CLEAR
067266  030000          .WORD   OPI!IVC                                 ;RELEASE
067270  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
067272  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
067274  020000          .WORD   OPI                                     ;READ IN PRESET
067276  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
067300  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
067302  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
067304  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
067306  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
067310  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
067312  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
067314  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
067316  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
067320  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
067322  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
067324  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
067326  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
067330  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
067332  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
067334  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
067336  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
067340  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
067342  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
067344  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
067346  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
067350  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
067352  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

067354  001     ATNTBL: .BYTE   1.
067355  002             .BYTE   2.
067356  004             .BYTE   4.
067357  010             .BYTE   8.
067360  020             .BYTE   16.
067361  040             .BYTE   32.
067362  100             .BYTE   64.
067363  200             .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

067364                  RGDTPT:
067364                  MIXED:
067364  000000                  .WORD   0.
067366  000001                  .WORD   1.
067370  000003                  .WORD   3.
067372  000007                  .WORD   7.
067374  000017                  .WORD   15.
067376  000037                  .WORD   31.
067400  000077                  .WORD   63.
067402  000177                  .WORD   127.
067404  000377                  .WORD   255.
067406  000777                  .WORD   511.
067410  001777                  .WORD   1023.
067412  003777                  .WORD   2047.
067414  007777                  .WORD   4095.
067416  017777                  .WORD   8191.
067420  037777                  .WORD   16383.
067422  077777                  .WORD   32767.
067424  177777          ONES:   .WORD   65535.
067426  177777                  .WORD   65535.
067430  077777                  .WORD   32767.
067432  037777                  .WORD   16383.
067434  017777                  .WORD   8191.
067436  007777                  .WORD   4095.
067440  003777                  .WORD   2047.
067442  001777                  .WORD   1023.
067444  000777                  .WORD   511.
067446  000377                  .WORD   255.
067450  000177                  .WORD   127.
067452  000077                  .WORD   63.
067454  000037                  .WORD   31.
067456  000017                  .WORD   15.
067460  000007                  .WORD   7.
067462  000003                  .WORD   3.
067464  000001                  .WORD   1.
067466  000000          ZEROS:  .WORD   0.
067470  000000                  .WORD   0.
067472  000001                  .WORD   1.
067474  000002                  .WORD   2.
067476  000004                  .WORD   4.
067500  000010                  .WORD   8.
067502  000020                  .WORD   16.
067504  000040                  .WORD   32.
067506  000100                  .WORD   64.
067510  000200                  .WORD   128.
067512  000400                  .WORD   256.
067514  001000                  .WORD   512.
067516  002000                  .WORD   1024.
067520  004000                  .WORD   2048.
067522  010000                  .WORD   4096.
067524  020000                  .WORD   8192.
067526  040000                  .WORD   16384.
067530  100000                  .WORD   32768.
067532  100000                  .WORD   32768.
```

```
067534  040000                          .WORD   16384.
067536  020000                          .WORD   8192.
067540  010000                          .WORD   4096.
067542  004000                          .WORD   2048.
067544  002000                          .WORD   1024.
067546  001000                          .WORD   512.
067550  000400                          .WORD   256.
067552  000200                          .WORD   128.
067554  000100                          .WORD   64.
067556  000040                          .WORD   32.
067560  000020                          .WORD   16.
067562  000010                          .WORD   8.
067564  000004                          .WORD   4.
067566  000002                          .WORD   2.
067570  000001                          .WORD   1.
067572  000000                          .WORD   0.
067574  177777                          .WORD   65535.
067576  177776                          .WORD   65534.
067600  177774                          .WORD   65532.
067602  177770                          .WORD   65528.
067604  177760                          .WORD   65520.
067606  177740                          .WORD   65504.
067610  177700                          .WORD   65472.
067612  177600                          .WORD   65408.
067614  177400                          .WORD   65280.
067616  177000                          .WORD   65024.
067620  176000                          .WORD   64512.
067622  174000                          .WORD   63488.
067624  170000                          .WORD   61440.
067626  160000                          .WORD   57344.
067630  140000                          .WORD   49152.
067632  100000                          .WORD   32768.
067634  000000                          .WORD   0.
067636  000000                          .WORD   0.
067640  100000                          .WORD   32768.
067642  140000                          .WORD   49152.
067644  160000                          .WORD   57344.
067646  170000                          .WORD   61440.
067650  174000                          .WORD   63488.
067652  176000                          .WORD   64512.
067654  177000                          .WORD   65024.
067656  177400                          .WORD   65280.
067660  177600                          .WORD   65408.
067662  177700                          .WORD   65472.
067664  177740                          .WORD   65504.
067666  177760                          .WORD   65520.
067670  177770                          .WORD   65528.
067672  177774                          .WORD   65532.
067674  177776                          .WORD   65534.
067676  177777                          .WORD   65535.
067700  125252                  EARLY:  .WORD   43690.
067702  152525                          .WORD   43690./2
067704  125252                          .WORD   43690.
067706  177777                          .WORD   65535.
067710  177776                          .WORD   65534.
067712  177775                          .WORD   65533.
```

```
067714  177773                          .WORD   65531.
067716  177767                          .WORD   65527.
067720  177757                          .WORD   65519.
067722  177737                          .WORD   65503.
067724  177677                          .WORD   65471.
067726  177577                          .WORD   65407.
067730  177377                          .WORD   65279.
067732  176777                          .WORD   65023.
067734  175777                          .WORD   64511.
067736  173777                          .WORD   63487.
067740  167777                          .WORD   61439.
067742  157777                          .WORD   57343.
067744  137777                          .WORD   49151.
067746  077777                          .WORD   32767.
067750  077777                          .WORD   32767.
067752  137777                          .WORD   49151.
067754  157777                          .WORD   57343.
067756  167777                          .WORD   61439.
067760  173777                          .WORD   63487.
067762  175777                          .WORD   64511.
067764  176777                          .WORD   65023.
067766  177377                          .WORD   65279.
067770  177577                          .WORD   65407.
067772  177677                          .WORD   65471.
067774  177737                          .WORD   65503.
067776  177757                          .WORD   65519.
070000  177767                          .WORD   65527.
070002  177773                          .WORD   65531.
070004  177775                          .WORD   65533.
070006  177776                          .WORD   65534.
070010  177777                          .WORD   65535.
070012                          ENRGDT:
```

```
                        .SBTTL  ERROR MESSAGE TABLE

070012  074406  000000          EMT1:   .WORD   EMS1,0
070016  074455  074472  000000  EMT2:   .WORD   EMS2,EMS3,0
070024  074455  074535  000000  EMT3:   .WORD   EMS2,EMS4,0
070032  074600  074630  000000  EMT4:   .WORD   EMS5,EMS6,0
070040  074600  074742  000000  EMT5:   .WORD   EMS5,EMS10,0
070046  101435  076623  000000  EMT6:   .WORD   EMS167,EMS64,0
070054  077371  101462  000000  EMT7:   .WORD   EMS110,EMS170,0
070062  074675  000000          EMT10:  .WORD   EMS7,0
070066  074742  000000          EMT11:  .WORD   EMS10,0
070072  075004  075015  000000  EMT12:  .WORD   EMS11,EMS12,0
070100  075056  075067  075100  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
070112  075152  076623  000000  EMT14:  .WORD   EMS17,EMS64,0
070120  075004  075235  000000  EMT15:  .WORD   EMS11,EMS21,0
070126  075004  075260  075407  EMT16:  .WORD   EMS11,EMS22,EMS27,0
070136  075004  075274  075420  EMT17:  .WORD   EMS11,EMS23,EMS30,0
070146  075004  075322  075420  EMT20:  .WORD   EMS11,EMS24,EMS30,0
070156  075004  075351  075407  EMT21:  .WORD   EMS11,EMS25,EMS27,0
070166  075004  075366  075407  EMT22:  .WORD   EMS11,EMS26,EMS27,0
070176  075004  075430  075420  EMT23:  .WORD   EMS11,EMS31,EMS30,0
070206  075004  075457  075420  EMT24:  .WORD   EMS11,EMS32,EMS30,0
070216  075004  075506  075420  EMT25:  .WORD   EMS11,EMS33,EMS30,0
070226  075004  075534  075420  EMT26:  .WORD   EMS11,EMS34,EMS30,0
070236  075004  075605  075420  EMT27:  .WORD   EMS11,EMS35,EMS30,0
070246  075004  075634  075420  EMT30:  .WORD   EMS11,EMS36,EMS30,0
070256  075004  075663  075420  EMT31:  .WORD   EMS11,EMS37,EMS30,0
070266  075004  075711  075420  EMT32:  .WORD   EMS11,EMS40,EMS30,0
070276  075004  075740  075420  EMT33:  .WORD   EMS11,EMS41,EMS30,0
070306  075004  075766  075420  EMT34:  .WORD   EMS11,EMS42,EMS30,0
070316  075004  076015  075420  EMT35:  .WORD   EMS11,EMS43,EMS30,0
070326  075004  076044  075420  EMT36:  .WORD   EMS11,EMS44,EMS30,0
070336  075004  076117  075420  EMT37:  .WORD   EMS11,EMS45,EMS30,0
070346  076707  075212  000000  EMT40:  .WORD   EMS66,EMS20,0
070354  077075  100603  077103  EMT41:  .WORD   EMS75,EMS141,EMS76,0
070364  100674  100704  077031  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
070376  076211  076327  077103  EMT43:  .WORD   EMS47,EMS53,EMS76,0
070406  077134  076327  077103  EMT44:  .WORD   EMS77,EMS53,EMS76,0
070416  077162  076327  077103  EMT45:  .WORD   EMS100,EMS53,EMS76,0
070426  077210  076327  077103  EMT46:  .WORD   EMS101,EMS53,EMS76,0
070436  076641  076623  000000  EMT47:  .WORD   EMS65,EMS64,0
070444  075056  075100  076575  EMT50:  .WORD   EMS13,EMS15,EMS63,0
070454  075004  076162  075420  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
070466  076211  076327  076757  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
070504  076211  076327  076757  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
070522  076240  076327  076757  EMT54:  .WORD   EMS50,EMS53,EMS67,0
070532  100631  100646  076327  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
070544  076267  077031  076757  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
070562  101435  077041  076757  EMT57:  .WORD   EMS167,EMS73,EMS67,0
070572  076412  076757  077571  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
070610  077016  076412  076757  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
070630  100562  076757  077571  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
070644  000000          EMT63:  .WORD
070646  100767  101032  077004  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
070666  076315  102062  102243  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
070706  101374  077265  077031  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
070720  101374  077265  077031  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
070732  076162  075212  101267  EMT70:  .WORD   EMS46,EMS20,EMS163,0
070742  077016  077210  101267  EMT71:  .WORD   EMS71,EMS101,EMS163,0
070752  076211  077031  101267  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
070770  076211  077031  101267  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
071006  076412  076327  101267  EMT74:  .WORD   EMS56,EMS53,EMS163,0
071016  077016  076412  101267  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
071036  076240  076327  101267  EMT76:  .WORD   EMS50,EMS53,EMS163,0
071046  100631  100646  076327  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
071060  077075  100603  101267  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
071076  077075  100767  101267  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
071114  101435  077041  101267  EMT102: .WORD   EMS167,EMS73,EMS163,0
071124  100752  101006  077056  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
071136  076267  077056  101267  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
071154  101374  077162  101267  EMT105: .WORD   EMS165,EMS100,EMS163,0
071164  101374  077134  101267  EMT106: .WORD   EMS165,EMS77,EMS163,0
071174  100674  100704  076327  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
071214  077371  077431  077576  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
071226  077515  074535  000000  EMT111: .WORD   EMS113,EMS4,0
071234  077515  074472  000000  EMT112: .WORD   EMS113,EMS3,0
071242  077371  077571  077576  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
071256  077515  077650  000000  EMT114: .WORD   EMS113,EMS120,0
071264  077665  100325  100351  EMT115: .WORD   EMS121,EMS132,EMS133,0
071274  077730  100325  100351  EMT116: .WORD   EMS122,EMS132,EMS133,0
071304  077765  100325  100351  EMT117: .WORD   EMS123,EMS132,EMS133,0
071314  100030  100325  100351  EMT120: .WORD   EMS124,EMS132,EMS133,
071324  100062  100325  100351  EMT121: .WORD   EMS125,EMS132,EMS133,0
071334  100125  100325  100351  EMT122: .WORD   EMS126,EMS132,EMS133,0
071344  100525  100325  100351  EMT123: .WORD   EMS137,EMS132,EMS133,0
071354  100227  100325  100351  EMT124: .WORD   EMS130,EMS132,EMS133,0
071364  100265  100325  100351  EMT125: .WORD   EMS131,EMS132,EMS133,0
071374  077665  100325  100374  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
071406  077730  100325  100374  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
071420  077765  100325  100374  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
071432  100030  100325  100374  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
071444  100062  100325  100374  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
071456  100125  100325  100374  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
071470  100525  100325  100374  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
071502  100227  100325  100374  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
071514  100265  100325  100374  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
071526  077665  100325  100436  EMT137: .WORD   EMS121,EMS132,EMS135,0
071536  077765  100325  100436  EMT140: .WORD   EMS123,EMS132,EMS135,0
071546  077665  100325  100500  EMT141: .WORD   EMS121,EMS132,EMS136,0
071556  100525  100325  100500  EMT142: .WORD   EMS137,EMS132,EMS136,0
071566  100030  100325  100500  EMT143: .WORD   EMS124,EMS132,EMS136,0
071576  100062  100325  100500  EMT144: .WORD   EMS125,EMS132,EMS136,0
071606  100125  100325  100500  EMT145: .WORD   EMS126,EMS132,EMS136,0
071616  100265  100325  100500  EMT146: .WORD   EMS131,EMS132,EMS136,0
071626  101234  100325  100500  EMT147: .WORD   EMS162,EMS132,EMS136,0
071636  100227  100325  100500  EMT150: .WORD   EMS130,EMS132,EMS136,0
071646  100562  077571  100603  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
071660  100631  077571  100646  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
071672  100674  100704  100733  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
071710  100674  100704  075212  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
071726  100767  101032  101100  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
071740  100752  101006  101100  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
072752  100752  101006  101134  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
071764  101204  101134  101167  EMT160: .WORD   EMS161,EMS156,EMS160,0
071774  075351  075407  101134  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
072006  075366  075407  101134  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
072020  076211  101267  100562  EMT163: .WORD   EMS47,EMS163,EMS140,0
072030  076211  075212  000000  EMT164: .WORD   EMS47,EMS20,0
072036  076412  075212  000000  EMT165: .WORD   EMS56,EMS20,0
072044  076162  075212  000000  EMT166: .WORD   EMS46,EMS20,0
072052  102563  075212  000000  EMT167: .WORD   EMS224,EMS20,0
072060  101435  101100  101410  EMT170: .WORD   EMS167,EMS154,EMS166,0
072070  101435  101100  101152  EMT171: .WORD   EMS167,EMS154,EMS157,0
072100  101435  101100  101114  EMT172: .WORD   EMS167,EMS154,EMS155,0
072110  101511  100325  100351  EMT173: .WORD   EMS171,EMS132,EMS133,0
072120  101511  100325  100374  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
072132  077515  101664  000000  EMT175: .WORD   EMS113,EMS177,0
072140  101706  101723  000000  EMT176: .WORD   EMS200,EMS201,0
072146  102021  100603  075420  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
072160  102021  076267  075420  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
072172  102021  102034  075420  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
072204  102021  076240  075420  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
072216  102021  100646  075420  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
072230  100767  077056  101771  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
072246  076240  077265  077031  EMT205: .WORD   EMS50,EMS103,EMS72,0
072256  077274  077041  101771  EMT206: .WORD   EMS104,EMS73,EMS202,0
072266  077075  100603  077571  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
072300  077075  100767  000000  EMT210: .WORD   EMS75,EMS150,0
072306  076267  077031  077571  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
072322  100631  076327  077571  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
072336  076240  077265  076327  EMT213: .WORD   EMS50,EMS103,EMS53,0
072346  076315  102062  101410  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
072366  076315  077626  076412  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
072400  076412  075420  076623  EMT216: .WORD   EMS56,EMS30,EMS64,0
072410  076162  075420  076623  EMT217: .WORD   EMS46,EMS30,EMS64,0
072420  075430  075420  076623  EMT220: .WORD   EMS31,EMS30,EMS64,0
072430  076211  075420  076623  EMT221: .WORD   EMS47,EMS30,EMS64,0
072440  076315  077626  077134  EMT222: .WORD   EMS52,EMS117,EMS77,0
072450  076315  077626  076575  EMT223: .WORD   EMS52,EMS117,EMS63,0
072460  000000                  EMT224: .WORD
072462  000000                  EMT225: .WORD
072464  000000                  EMT226: .WORD
072466  000000                  EMT227: .WORD
072470  000000                  EMT230: .WORD
072472  000000                  EMT231: .WORD
072474  000000                  EMT232: .WORD
072476  000000                  EMT233: .WORD
072500  000000                  EMT234: .WORD
072502  000000                  EMT235: .WORD
072504  000000                  EMT236: .WORD
072506  000000                  EMT237: .WORD
072510  000000                  EMT240: .WORD
072512  000000                  EMT241: .WORD
072514  000000                  EMT242: .WORD
072516  000000                  EMT243: .WORD
072520  000000                  EMT244: .WORD
072522  000000                  EMT245: .WORD
072524  101435  100325  102121  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
072534  101435  100325  102146  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
072546  101435  102157  102146  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
072562  102175  102220  000000  EMT251: .WORD   EMS212,EMS213,0
072570  101374  102175  000000  EMT252: .WORD   EMS165,EMS212,0
072576  100752  101006  101134  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
072614  102021  077210  075420  EMT254: .WORD   EMS203,EMS101,EMS30,0
072624  102021  101435  075420  EMT255: .WORD   EMS203,EMS167,EMS30,0
072634  102021  077162  075420  EMT256: .WORD   EMS203,EMS100,EMS30,0
072644  075004  076162  075420  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
072656  076315  077626  076412  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
072670  076315  102062  102365  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
072710  077274  077041  101771  EMT262: .WORD   EMS104,EMS73,EMS202,0
072720  076240  076327  077236  EMT263: .WORD   EMS50,EMS53,EMS102,0
072730  076412  076327  077236  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
072750  077016  076412  077236  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
072770  100631  100646  076327  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
073002  076240  100733  077571  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
073020  076211  076327  077236  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
073034  076211  076327  077236  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
073052  077075  100603  077236  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
073070  076267  077056  077236  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
073106  076575  076327  076467  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
073124  077576  076327  075740  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
073136  076211  076327  076467  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
073152  076211  076327  076467  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
073170  076412  076327  076467  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
073210  077016  076412  076327  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
073232  101374  077162  077265  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
073244  101374  077210  077265  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
073256  101374  077134  077265  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
073270  076162  075420  076623  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
073302  076240  076327  076467  EMT306: .WORD   EMS50,EMS53,EMS57,0
073312  076240  100733  077571  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
073332  100631  100646  076327  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
073344  077274  077265  076327  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
073356  077323  077265  076327  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
073370  100674  100704  077265  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
073410  075766  077265  076327  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
073422  075430  077265  076327  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
073434  077016  075430  077265  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
073446  076015  077265  076467  EMT317: .WORD   EMS43,EMS103,EMS57,0
073456  076117  077265  076467  EMT320: .WORD   EMS45,EMS103,EMS57,0
073466  076044  077265  076467  EMT321: .WORD   EMS44,EMS103,EMS57,0
073476  077352  075212  000000  EMT322: .WORD   EMS106,EMS20,0
073504  075634  077265  076467  EMT323: .WORD   EMS36,EMS103,EMS57,0
073514  101564  075634  077265  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
073526  101544  075634  077265  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
073540  075056  101601  075100  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
073556  100752  101006  077056  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
073570  076707  076327  101607  EMT330: .WORD   EMS66,EMS53,EMS175,0
073600  075506  077265  076327  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
073612  075711  077265  076327  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
073624  076267  077056  076467  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
073642  077075  100603  076467  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
073660  077075  100767  101032  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
073700  076515  076530  076557  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
073710  077576  077626  075534  EMT337: .WORD   EMS116,EMS117,EMS34,0
073720  075534  076327  076341  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
073732  076363  075534  000000  EMT341: .WORD   EMS55,EMS34,0
073740  076315  077626  076412  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
073752  076315  077626  076117  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
073764  076315  077626  076044  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
073776  076315  077626  102405  EMT345: .WORD   EMS52,EMS117,EMS221,0
074006  077352  075212  077571  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
074022  076315  102062  102455  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
074034  077075  100767  077236  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
074052  101435  077041  077236  EMT351: .WORD   EMS167,EMS73,EMS102,0
074062  102261  000000          EMT352: .WORD   EMS215,0
074066  102332  102021  102302  EMT353: .WORD   EMS217,EMS203,EMS216,0
074076  102405  075212  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
074104  102635  103441  103516  EHT1:   .WORD   EH1,STSH1,STSH2,STSH4,0
074116  103441  103516  103643  EHT2:   .WORD   STSH1,STSH2,STSH4,0

074126  102654  000000          EHT110: .WORD   EH110,0
074132  102663  000000          EHT111: .WORD   EH111,0

074136  102702  000000          EHT114: .WORD   EH114,0
074142  102731  103441  103516  EHT223: .WORD   EH223,STSH1,STSH2,STSH4,0
074154  102757  103441  103516  EHT256: .WORD   EH256,STSH1,STSH2,STSH4,0

074166  103034  103441  103516  EHT336: .WORD   EH336,STSH1,STSH2,STSH4,0
074200  103073  103441  103516  EHT337: .WORD   EH337,STSH1,STSH2,STSH4,0
074212  103232  103441  103516  EHT344: .WORD   EH344,STSH1,STSH2,STSH4,0

074224  103372  000000          EHT353: .WORD   EH353,0
```

```
074230  103702  103776  104014  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
074240  103776  104014  104046  EDT2:    .WORD   STSD1,STSD2,STSD4

074246  103710                  EDT110:  .WORD   ED110
074250  103714                  EDT111:  .WORD   ED111

074252  103722                  EDT114:  .WORD   ED114
074254  103732  103776  104014  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4

074264  103742  103776  104014  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
074274  103754  103776  104014  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
074304  103754  103776  104014  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0

074316  103766                  EDT353:  .WORD   ED353
```

```
074320  104061  104077  104077  EFT1:   .WORD   EF111,STSF,STSF,STSF
074330  104077  104077  104077  EFT2:   .WORD   STSF,STSF,STSF

074336  104060                  EFT110: .WORD   EF110
074340  104061                  EFT111: .WORD   EF111

074342  104063                  EFT114: .WORD   EF114
074344  104063  104077  104077  EFT223: .WORD   EF114,STSF,STSF,STSF

074354  104066  104077  104077  EFT336: .WORD   EF336,STSF,STSF,STSF
074364  104066  104077  104077  EFT337: .WORD   EF336,STSF,STSF,STSF
074374  104066  104077  104077  EFT344: .WORD   EF336,STSF,STSF,STSF

074404  104063                  EFT353: .WORD   EF114
```

```
                                .SBTTL  ERROR MESSAGE STRINGS

074406 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
074455    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
074472 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
074535    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
074600 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED, @
074630 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
074675    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
074742 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
075004 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
075015    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
075056 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
075067    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
075100 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
075122 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
075152 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
075212 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
075235    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
075260 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
075274 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
075322 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
075351    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
075366 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
075407    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
075420 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
075430 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
075457    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
075506 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
075534 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
075605    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
075634 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
075663    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
075711    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
075740 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
075766 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
076015    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
076044 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
076117    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
076162 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
076211    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
076240 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
076267    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
076315    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
076327    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
076341    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
076363    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
076412 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
076467    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
076515    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
076530 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
076557    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
076575    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
076623    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
076641    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
076707    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
076757     104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
077004  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
077016  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
077031     111  020123  042523  EMS72:  .ASCIZ  @IS SET @
077041     104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
077056  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
077075     114  051517  020124  EMS75:  .ASCIZ  @LOST @
077103     104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
077134  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
077162  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
077210  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
077236  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
077265     105  051122  051117  EMS103: .ASCIZ  @ERROR @
077274  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
077323     042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
077352  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
077371     102  051526  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
077420  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
077431     127  042510  020116  EMS112: .ASCIZ  @WHEN READING/WRITING RH REGISTERS @
077473     101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
077515     124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
077545     116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
077571     040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
077576  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
077626  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
077650  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
077665     103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
077730  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
077765     103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
100030  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
100062  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
100125     115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
100170  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
100227     105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
100265     115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
100325     116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
100351     125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
100374  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E. BIT 5 OF @
100436  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
100500  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
100525     104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
100562  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
100603     042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
100631     104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
100646  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
100674  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
100704  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
100733     123  047510  046122  EMS146: .ASCIZ  @SHOULD BE SET @
100752  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
100767     040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
101006  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
101032  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
101056  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
101100  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
101114  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
101134  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
101152  052122  020103  047503  EMS157: .ASCIZ  @RTC COMMAND @
101167     122  050111  041440  EMS160: .ASCIZ  @RIP COMMAND @
101204  043117  051506  052105  EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
101234  051105  047522  020122  EMS162: .ASCIZ  @ERROR REGISTER #2, RMER2, @
101267     104  051125  047111  EMS163: .ASCIZ  @DURING RECALIBRATE @
101313     111  020123  047111  EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
101362  054503  044514  042116          .ASCIZ  @CYLINDER @
101374  047125  054105  042520  EMS165: .ASCIZ  @UNEXPECTED @
101410  042522  040503  044514  EMS166: .ASCIZ  @RECALIBRATE COMMAND @
101435     042  052101  021101  EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
101462  044127  047105  051040  EMS170: .ASCIZ  @WHEN READING REGISTER @
101511     105  051122  051117  EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
101544  047516  051116  041505  EMS172: .ASCIZ  @NONRECOVERABLE @
101564  042522  047503  042526  EMS173: .ASCIZ  @RECOVERABLE @
101601     104  052101  020101  EMS174: .ASCIZ  @DATA @
101607     104  051125  047111  EMS175: .ASCIZ  @DURING WRITE COMMAND @
101635     042  050117  021105  EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
101664  047111  053440  044522  EMS177: .ASCIZ  @IN WRITE PROTECT @
101706  040503  020116  047516  EMS200: .ASCIZ  @CAN NOT SET @
101723     104  040511  047107  EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
101771     104  051125  047111  EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
102021     111  041516  051117  EMS203: .ASCIZ  @INCORRECT @
102034  053442  046122  020042  EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
102062  054105  041505  052125  EMS205: .ASCIZ  @EXECUTED @
102074  044527  044124  041440  EMS206: .ASCIZ  @WITH COMP ERROR SET @
102121     042  047507  020042  EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
102146  051127  052111  047111  EMS210: .ASCIZ  @WRITING @
102157     127  051501  051040  EMS211: .ASCIZ  @WAS RESET BY @
102175     120  047522  051107  EMS212: .ASCIZ  @PROGRAM INTERRUPT @
102220  040527  020123  047516  EMS213: .ASCIZ  @WAS NOT GENERATED @
102243     123  042505  020113  EMS214: .ASCIZ  @SEEK COMMAND @
102261     120  047522  051107  EMS215: .ASCIZ  @PROGRAM TIMEOUT @
102302  052504  044522  043516  EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
102332  047514  045517  040440  EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
102365     123  040505  041522  EMS220: .ASCIZ  @SEARCH COMMAND @
102405     102  042101  051440  EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
102455     101  042040  052101  EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
102506  042510  042101  051105  EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
102563     116  047117  054105  EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
102635    105  050130  052103  EH1:    .ASCIZ  @EXPCTD  RECEVD@
102654 052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
102663    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

102702 042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
102731    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
102757    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
103006 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

103034 042107  042101  051522  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
103073    122  041515  031123  EH337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
103133    137  057537  057537          .ASCII  @++++++++++++++++  ++++++++++++++++@<CR><LF>
103173    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

103232 046522  051105  020061  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
103273    137  057537  057537          .ASCII  @++++++++++++++++  WORD     BIT@<CR><LF>
103332 054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
103372 054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
103412 051040  046115  020101          .ASCIZ  @ RMLA    RMLA    RMOF @

103441    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
103506 020040  051040  040515          .ASCIZ  @    RMAS@
103516 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
103563    040  020040  051040          .ASCIZ  @     RMEC1   RMEC2@
103605    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
103643    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
```

```
                103702                          .EVEN
103702  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
103710  001276  000000          ED110:  .WORD   $BASE,0
103714  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

103722  001354  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
103732  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

103742  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

103754  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
103766  001140  001142  001430  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

103776  001326  001336  001340  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
104014  001330  001332  001334  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
104030  001374  000000          .WORD   RMEC2I,0
104034  001334  001362  001360  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
104046  001352  001366  001354  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
104060  000             EF110:  .BYTE   0
104061  000  000        EF111:  .BYTE   0,0
104063  000  000  000   EF114:  .BYTE   0,0,0
104066  000  000  000   EF336:  .BYTE   0,0,0,0
104072  000  000  000   EF337:  .BYTE   0,0,0,0,0

104077  000  000  000   STSF:   .BYTE   0,0,0,0,0,0,0
```

```
104106  000402
105112  177777
105114  177777
105116  000402
106122  177777
106124  177777
```

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

```
                                BUFFER:
106126                          BUFONE: .BLKW   258.
106126  000402                  BUFTWO: .BLKW   258.
107132  000402

        106126                  .       =       BUFFER
106126                          HELP:
106126  005015                  .ASCII  <CR><LF>
106126  046011  051511  020124  .ASCII  @       LIST OF TESTS@<CR><LF>
106150  057411  057537  057537  .ASCII  @       ++++++++++++++@<CR><LF>
106170  030524  041411  047117  .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
106223     124  004462  042504  .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
106255     124  004463  051104  .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
106301     124  004464  047125  .ASCII  @T4     UNIBUS INITIALIZE TEST@<CR><LF>
106334  032524  041411  047117  .ASCII  @T5     CONTROLLER CLEAR TEST@<CR><LF>
106366  033124  042411  051122  .ASCII  @T6     ERROR CLEAR TEST@<CR><LF>
106413     124  004467  051104  .ASCII  @T7     DRIVE STATUS TEST@<CR><LF>
106441     124  030061  050011  .ASCII  @T10    PRIMARY/SECONDARY ERROR TEST@<CR><LF>
106503     124  030461  042011  .ASCII  @T11    DIAGNOSTIC MODE TEST@<CR><LF>
106535     124  031061  050011  .ASCII  @T12    PACK ACKNOWLEDGE TEST@<CR><LF>
106570  030524  004463  042522  .ASCII  @T13    RECALIBRATE TEST@<CR><LF>
106616  030524  004464  041101  .ASCII  @T14    ABORT RECALIBRATE TEST@<CR><LF>
106652  030524  004465  053111  .ASCII  @T15    IVC RECALIBRATE TEST@<CR><LF>
106704  030524  004466  040511  .ASCII  @T16    IAE RECALIBRATE TEST@<CR><LF>
106736  030524  004467  042522  .ASCII  @T17    RECALIBRATE AT OFFSET@<CR><LF>
106771     124  030062  042011  .ASCII  @T20    DRIVE CLEAR TEST@<CR><LF>
107017     124  030462  047011  .ASCII  @T21    NOP TEST@<CR><LF>
107035     124  031062  047411  .ASCII  @T22    OFFSET TEST@<CR><LF>
107056  031124  004463  047507  .ASCII  @T23    GO/ATA TEST@<CR><LF>
107077     124  032062  053411  .ASCII  @T24    WRITE ATA TEST@<CR><LF>
107123     124  032462  042411  .ASCII  @T25    ERROR/ATA TEST@<CR><LF>
107147     124  033062  050011  .ASCII  @T26    PROGRAM INTERRUPT TEST@<CR><LF>
107203     124  033462  044411  .ASCII  @T27    INHIBIT INTERRUPT TEST@<CR><LF>
107237     124  030063  051011  .ASCII  @T30    RETURN TO CENTERLINE TEST@<CR><LF>
107276  031524  004461  042522  .ASCII  @T31    READ IN PRESET TEST@<CR><LF>
107327     124  031063  051011  .ASCII  @T32    RMDC CLEAR OFFSET TEST@<CR><LF>
107363     124  031463  044411  .ASCII  @T33    ILLEGAL FUNCTION TEST@<CR><LF>
107416  031524  004464  047111  .ASCII  @T34    INVALID COMMAND TEST@<CR><LF>
107450  031524  004465  047111  .ASCII  @T35    INVALID ADDRESS ERROR TEST@<CR><LF>
107510  031524  004466  051127  .ASCII  @T36    WRITE LOCK ERROR TEST@<CR><LF>
107543     124  033463  047411  .ASCII  @T37    OPI TEST@<CR><LF>
107561     124  030064  042411  .ASCII  @T40    ERROR ABORT TESTS@<CR><LF>
107610  032124  004461  046522  .ASCII  @T41    RMR TEST@<CR><LF>
107626  032124  004462  040520  .ASCII  @T42    PARITY ERROR TEST@<CR><LF>
107655     124  031464  044411  .ASCII  @T43    ILLEGAL REGISTER TEST@<CR><LF>
107710  032124  004464  042523  .ASCII  @T44    SEEK LAST CYLINDER@<CR><LF>
107740  032124  004465  042523  .ASCII  @T45    SEEK FIRST CYLINDER@<CR><LF>
107771     124  033064  051411  .ASCII  @T46    SEEK PRIME CYLINDERS@<CR><LF>
110023     124  033464  051411  .ASCII  @T47    SEEK ZERO DIFFERENCE@<CR><LF>
110055     124  030065  051411  .ASCII  @T50    SEEK MAXIMUM DIFFERENCE FORWARD@<CR><LF>
110122  032524  004461  042523  .ASCII  @T51    SEEK ADJACENT FORWARD@<CR><LF>
110155     124  031065  051411  .ASCII  @T52    SEEK ADJACENT REVERSE@<CR><LF>
110210  032524  004463  042523  .ASCII  @T53    SEEK INVALID SECTOR@<CR><LF>
110241     124  032065  051411  .ASCII  @T54    SEEK INVALID TRACK@<CR><LF>
110271     124  032465  051411  .ASCII  @T55    SEEK INVALID CYLINDER@<CR><LF>
110324  032524  004466  053111  .ASCII  @T56    IVC SEEK TEST@<CR><LF>
```

```
110347     124  033465  040411  .ASCII  @T57     ABORT SEEK TEST@<CR><LF>
110374  033124  004460  042523  .ASCII  @T60     SEEK AT OFFSET@<CR><LF>
110420  033124  004461  047514  .ASCII  @T61     LOOK AHEAD TEST@<CR><LF>
110445     124  031066  051411  .ASCII  @T62     SEARCH ON CYLINDER@<CR><LF>
110475     124  031466  051411  .ASCII  @T63     SEARCH OFF CYLINDER@<CR><LF>
110526  033124  004464  042523  .ASCII  @T64     SEARCH INVALID SECTOR@<CR><LF>
110561     124  032466  051411  .ASCII  @T65     SEARCH INVALID TRACK@<CR><LF>
110613     124  033066  051411  .ASCII  @T66     SEARCH INVALID CYLINDER@<CR><LF>
110650  033124  004467  053111  .ASCII  @T67     IVC SEARCH TEST@<CR><LF>
110675     124  030067  040411  .ASCII  @T70     ABORT SEARCH TEST@<CR><LF>
110724  033524  004461  042523  .ASCII  @T71     SEARCH AT OFFSET@<CR><LF>
110752  033524  004462  042510  .ASCII  @T72     HEAD ALIGNMENT SEEK @<CR><LF>
111004  005015                  .ASCII  <CR><LF>
111006  047411  042520  040522  .ASCII  @        OPERATIONAL SWITCH SETTINGS@<CR><LF>
111044  057411  057537  057537  .ASCII  @        ++++++++++++++++++++++++++@<CR><LF>
111102  005015                  .ASCII  <CR><LF>
111104  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
111122  026455  026455  026455  .ASCII  @------                  ----------@<CR><LF>
111160  020040  032461  004411  .ASCII  @   15                   HALT ON ERROR@<CR><LF>
111205     040  030440  004464  .ASCII  @   14                   LOOP ON TEST@<CR><LF>
111231     040  030440  004463  .ASCII  @   13                   INHIBIT ERROR TYPEOUTS@<CR><LF>
111267     040  030440  004462  .ASCII  @   12                   @<CR><LF>
111277     040  030440  004461  .ASCII  @   11                   INHIBIT ITERATIONS@<CR><LF>
111331     040  030440  004460  .ASCII  @   10                   BELL ON ERROR@<CR><LF>
111356  020040  034440  004411  .ASCII  @    9                   LOOP ON ERROR@<CR><LF>
111403     040  020040  004470  .ASCII  @    8                   LOOP ON TEST IN SWR<7:0>@<CR><LF>
111443     040  020040  004467  .ASCII  @    7                   TN128@<CR><LF>
111460  020040  033040  004411  .ASCII  @    6                   TN64@<CR><LF>
111474  020040  032440  004411  .ASCII  @    5                   TN32@<CR><LF>
111510  020040  032040  004411  .ASCII  @    4                   TN16@<CR><LF>
111524  020040  031440  004411  .ASCII  @    3                   TN8@<CR><LF>
111537     040  020040  004462  .ASCII  @    2                   TN4@<CR><LF>
111552  020040  030440  004411  .ASCII  @    1                   TN2@<CR><LF>
111565     040  020040  004460  .ASCII  @    0                   TN1@<CR><LF>
111600  005015                  .ASCII  <CR><LF>
111602  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1945#  2053   2094
ACDW1 = 000000          2053   2096
ACDW2 = 000000          2053   2097
ACKSTS  054356          4912   9417   11028#
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          4215#  4216   4257#  4258#  4259   4301#  4302#  4303   4340#  4341#  4343#  4474#  4475#
                        4476   4551#  4552#  4553   4598#  4599#  4600   4664#  4665#  4666   4711#  4712#  4713
                        4858#  4859#  4860   4932#  4933#  4935#  4990#  4991#  4993#  5066#  5067#  5069#  5138#
                        5139#  5141#  5201#  5202#  5204#  5272#  5273#  5274   5335#  5336#  5337   5398#  5399#
                        5400   5473#  5474#  5475   5572#  5573#  5574   5650#  5651#  5652   5733#  5734#  5735
                        5848#  5849#  5850   5922#  5923#  5924   5998#  5999#  6000   6084#  6085#  6086   6153#
                        6154#  6155   6268#  6269#  6270   6377#  6378#  6379   6492#  6493#  6494   6607#  6608#
                        6609   6731#  6732#  6733   6807#  6808#  6809   6901#  6902#  6903   6959#  6960#  6961
                        7057#  7058#  7060#  7130#  7131#  7133#  7203#  7204#  7206#  7283#  7284#  7286#  7363#
                        7364#  7366#  7444#  7445#  7447#  7524#  7525#  7527#  7603#  7604#  7605   7699#  7700#
                        7701   7798#  7799#  7800   7897#  7898#  7899   7976#  7977#  7978   8058#  8059#  8061#
                        8142#  8143#  8144   8243#  8244#  8245   8352#  8353#  8354   8465#  8466#  8467   8561#
                        8562#  8563   8648#  8649#  8650   8742#  8743#  8744   9822#  8823#  8824   8904#  8905#
                        8906   8988#  8989#  8991#
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   10198  10213  10228  10243  10247  10494  10498  12989
```

```
APASS = 000000  2053  2058
APE   = 100000  1924#
APRIOR= 000000  2053
APTCSU= 000040  12307  12987#
APTENV= 000001  12300  12521  12943  12985#
APTSIZ= 000200  3960  12984#
APTSPO= 000100  12302  12945  12986#
ARGS  = 000003  4531#  4539#  4693#  4700#  4905#  4912#  4920#  4942#  4965#  4972#  4979#  5001#  5041#
                5048#  5055#  5076#  5117#  5127#  5148#  5176#  5183#  5190#  5212#  5237#  5239#  5241#
                5247#  5254#  5261#  5278#  5309#  5316#  5323#  5341#  5363#  5370#  5379#  5386#  5404#
                5427#  5446#  5455#  5462#  5480#  5508#  5532#  5545#  5554#  5561#  5579#  5624#  5637#
                5657#  5715#  5722#  5740#  5795#  5814#  5821#  5830#  5837#  5855#  5928#  5951#  5970#
                5979#  5986#  6005#  6035#  6057#  6066#  6073#  6091#  6113#  6160#  6234#  6275#  6343#
                6384#  6458#  6499#  6573#  6614#  6697#  6738#  6745#  6748#  6778#  6784#  6785#  6791#
                6814#  6821#  6850#  6884#  6908#  6917#  6933#  6939#  6966#  6988#  7000#  7068#  7096#
                7103#  7110#  7119#  7141#  7169#  7176#  7183#  7192#  7215#  7241#  7248#  7255#  7264#
                7294#  7324#  7331#  7338#  7347#  7374#  7404#  7411#  7418#  7427#  7455#  7484#  7491#
                7498#  7507#  7535#  7564#  7571#  7578#  7587#  7611#  7641#  7654#  7660#  7667#  7675#
                7684#  7706#  7736#  7755#  7762#  7770#  7779#  7805#  7835#  7854#  7861#  7869#  7878#
                7942#  7956#  7965#  7983#  8024#  8031#  8038#  8047#  8069#  8101#  8109#  8116#  8123#
                8132#  8184#  8249#  8279#  8286#  8306#  8313#  8320#  8329#  8358#  8390#  8397#  8417#
                8424#  8431#  8440#  8472#  8502#  8515#  8522#  8531#  8567#  8598#  8612#  8621#  8655#
                8685#  8698#  8705#  8714#  8749#  8789#  8803#  8812#  8829#  8870#  8877#  8884#  8893#
                8910#  8949#  8956#  8963#  8972#  9011#  9039#  9046#  9053#  9062#
ASNDA   001476  2176#
ASNDC   001474  2175#
ASWREG= 000000  2053  2066
ATA   = 100000  1705#  4620  5439  5442  5443  5501  5503  5505  5538  5541  5600  5603  5604
                5630  5633  5678  5681  5682  5708  5710  5712  5963  5966  5967  6251  6252
                6254  6360  6361  6363  6475  6476  6478  6590  6591  6593  6714  6715  6717
                10025  10026  10028  10760  10761  10796  10799  11291  11292  11331  11334  11755  11756
                11808  11811  12989
ATESTN= 000000  2053  2057
ATNMSK= 000377  1742#  5607  11488
ATNTBL  067354  4158  4176  12989#
AUNIT = 000000  2053  2060
AUSWR = 000000  2053  2067
AVECT1= 120254  1946#  2053  2092
AVECT2= 000000  2053  2093
A16   = 000400  1895#  7037
A17   = 001000  1894#  7037
BACK  = 000001  4531#  4535  4539#  4543  4693#  4697  4700#  4704  4905#  4909  4912#  4916  4920#
                4924  4942#  4947  4965#  4969  4972#  4976  4979#  4983  5001#  5006  5041#  5045
                5048#  5052  5055#  5059  5076#  5081  5117#  5121  5127#  5131  5148#  5153  5176#
                5180  5183#  5187  5190#  5194  5212#  5217  5237#  5239#  5241#  5245  5247#  5251
                5254#  5258  5261#  5265  5278#  5283  5309#  5313  5316#  5320  5323#  5327  5341#
                5346  5363#  5367  5370#  5373#  5376  5379#  5383  5386#  5390  5404#  5409  5427#
                5431  5446#  5449#  5452  5455#  5459  5462#  5466  5480#  5485  5508#  5511#  5514
                5532#  5536  5545#  5548#  5551  5554#  5558  5561#  5565  5579#  5584  5624#  5628
                5637#  5640#  5643  5657#  5662  5715#  5719  5722#  5726  5740#  5745  5795#  5799
                5814#  5818  5821#  5824#  5827  5830#  5834  5837#  5841  5855#  5860  5928#  5933
                5951#  5955  5970#  5973#  5976  5979#  5983  5986#  5990  6005#  6010  6035#  6039
                6057#  6060#  6063  6066#  6070  6073#  6077  6091#  6096  6113#  6117  6160#  6165
                6234#  6238  6275#  6280  6343#  6347  6384#  6389  6458#  6462  6499#  6504  6573#
                6577  6614#  6619  6697#  6701  6738#  6743  6745#  6748#  6752  6778#  6782  6784#
                6785#  6789  6791#  6795  6814#  6819  6821#  6850#  6854  6884#  6888  6908#  6913
```

```
                    6917#   6921    6933#   6937    6939#   6943    6966#   6971    6988#   6992    7000#   7004    7068#
                    7073    7096#   7100    7103#   7107    7110#   7113#   7116    7119#   7123    7141#   7146    7169#
                    7173    7176#   7180    7183#   7186#   7189    7192#   7196    7215#   7220    7241#   7245    7248#
                    7252    7255#   7258#   7261    7264#   7268    7294#   7299    7324#   7328    7331#   7335    7338#
                    7341#   7344    7347#   7351    7374#   7379    7404#   7408    7411#   7415    7418#   7421#   7424
                    7427#   7431    7455#   7460    7484#   7488    7491#   7495    7498#   7501#   7504    7507#   7511
                    7535#   7540    7564#   7568    7571#   7575    7578#   7581#   7584    7587#   7591    7611#   7616
                    7641#   7645    7654#   7658    7660#   7664    7667#   7671    7675#   7678#   7681    7684#   7688
                    7706#   7711    7736#   7740    7755#   7759    7762#   7766    7770#   7773#   7776    7779#   7783
                    7805#   7810    7835#   7839    7854#   7858    7861#   7865    7869#   7872#   7875    7878#   7882
                    7942#   7946    7956#   7959#   7962    7965#   7969    7983#   7988    8024#   8028    8031#   8035
                    8038#   8041#   8044    8047#   8051    8069#   8074    8101#   8109#   8113    8116#   8120    8123#
                    8126#   8129    8132#   8136    8184#   8188    8249#   8254    8279#   8283    8286#   8290    8306#
                    8310    8313#   8317    8320#   8323#   8326    8329#   8333    8358#   8363    8390#   8394    8397#
                    8401    8417#   8421    8424#   8428    8431#   8434#   8437    8440#   8444    8472#   8477    8502#
                    8506    8515#   8519    8522#   8525#   8528    8531#   8535    8567#   8572    8598#   8602    8612#
                    8615#   8618    8621#   8625    8655#   8660    8685#   8689    8698#   8702    8705#   8708#   8711
                    8714#   8718    8749#   8754    8789#   8793    8803#   8806#   8809    8812#   8816    8829#   8834
                    8870#   8874    8877#   8881    8884#   8887#   8890    8893#   8897    8910#   8915    8949#   8953
                    8956#   8960    8963#   8966#   8969    8972#   8976    9011#   9016    9039#   9043    9046#   9050
                    9053#   9056#   9059    9062#   9066
BAI     = 000010    1914#   10414
BB00    = 000001    1832#
BB01    = 000002    1831#
BB02    = 000004    1830#
BB03    = 000010    1829#
BB04    = 000020    1828#
BB05    = 000040    1827#
BB06    = 000100    1826#
BB07    = 000200    1825#
BB08    = 000400    1824#
BB09    = 001000    1823#
BIT0    = 000001    1621#   4005    4325    4327    7041    9701
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    12387
BIT09   = 001000    1602#   1612    1690    1711    1725    1747    1764    1782    1823    1894    1907    12405   12532
BIT1    = 000002    1620#   7041    10431
BIT10   = 002000    1601#   1689    1710    1724    1746    1763    1781    1807    1822    1842    1893    1906    1929
                    12509
BIT11   = 004000    1600#   1641    1709    1723    1762    1780    1798    1806    1821    1841    1905    1928    9429
                    12412
BIT12   = 010000    1599#   1708    1722    1761    1779    1805    1820    1840    1904    1927    9379
BIT13   = 020000    1598#   1707    1721    1760    1778    1797    1819    1839    1891    1903    1926    12516
BIT14   = 040000    1597#   1706    1720    1759    1777    1796    1818    1838    1849    1890    1902    1925    9344
                    12373
BIT15   = 100000    1596#   1705    1719    1758    1776    1795    1817    1837    1848    1889    1901    1924    9331
BIT2    = 000004    1619#   9701
BIT3    = 000010    1618#   9457
BIT4    = 000020    1617#   9407
```

```
BIT5   = 000040          1616#
BIT6   = 000100          1615#  9356
BIT7   = 000200          1614#  4000   9707
BIT8   = 000400          1613#
BIT9   = 001000          1612#
BOTADR  043154           9195*  9213*  9216   9231   9276#
BOTFLG  043156           9181*  9223*  9226   9229*  9277#
BPTVEC= 000014           1628#
BSE    = 100000          1837#  10318
BUFFER  106126           8191   8211   12989#
BUFONE  106126           6209   6318   6433   6548   6672   6765   12989#
BUFTWO  107132          12989#
CC     = 004000          1821#
CH     = 002000          1822#
CHRCNT  043157           9182*  9200*  9206*  9207   9210*  9214   9219*  9230*  9278#
CKSWR  = 104410          12371  12505  12531  12882#
CLEAN  = ****** U        12989
CLKADR  001500           2177#  9663*  9665   9669*  9671   9696
CLKVCT  001502           2178#  9664*  9670*
CLR    = 000040          1912#  4007   10857
CLRSTS  053476           4531   7641   7736   7835   9366   10893#
CLSPRN  066306           4078   4094   4115   12989#
CMNSTA  006754           4030   4165#
CMPERR  061566           5370   5446   5508   5545   5637   5821   5970   6057   7110   7183   7255   7338   7418
                         7498   7578   7675   7770   7869   7956   8038   8123   8320   8431   8522   8612   8705
                         8803   8884   8963   9053   11980#
CNSL00  066452           4040   12989#
CNSL01  066511           4075   12989#
CNSL02  066536           4084   12989#
CNSL03  066617           4089   12989#
CNSL04  066647           4100   12989#
CNSL05  066723           4110   12989#
CNSL06  066757           4121   12989#
CNSL07  067004           4134   12989#
CNTCLR  053360           4264   4308   4352   4481   4519   4558   4605   4671   4718   4865   6169   6284   6393
                         6508   6623   6748   6917   6980   7629   7724   7823   7904   8149   8539   8634   8727
                         9346   10849#
CONT   = 000100          1785#
CR     = 000015          1536#  4148   9198   12059  12346  12356  12989
CRLF   = 000200          1537#  3984   12317  12356
CYLMSK= 001777           1812#
DBCK   = 100000          1758#
DBEN   = 040000          1759#  4372   4501   6859
DBL    = 002000          1929#
DCK    = 100000          1719#  1736   10356  10376
DDISP  = 177570          1543#  2029   3948
DEBL   = 020000          1760#
DEVSEL  051706           6988   9333   10559#
DISPLA  001156           2029#  3948*  3956*  12427*  12508*
DISPRE  000174           1956#  3956
DLT    = 100000          1901#  10200
DMD    = 000001          1772#  1791#  4372   4501   4726   4738   4740   4777   4789   4791   4873   5085   6180
                         6189   6295   6304   6404   6413   6519   6528   6634   6643   6663   6825   6834   6859
                         7912   8758
DPE    = 000010          1844#  5372   5448   5547   5823   5972   6059   7112   7185   7257   7340   7420   7500
                         7580   7677   7772   7871   7958   8040   8125   8322   8433   8524   8614   8707   8805
```

```
                    8886    8965    9055    9862   10142   10638   11177   11607
DPEHI  = 040000     1925#
DPELO  = 020000     1926#
DPR    = 000400     1712#   4461    4462    4648   10799   10996   10997   11463
DRQ    = 004000     1798#
DRVCLR= 000010      1656#   4834    5286   11451
DRVSTS  056714      5316   11442#
DRY    = 000200     1713#   4461    4462    4648    9829    9925    9928    9929   10799   10996   10997   11463
DSWR   = 177570     1542#   2028    3947
DTE    = 010000     1722#   1736
DTO    = 010000     1761#
DULPRT= 024024      1801#   4323    4327    4330
DVA    = 004000     1641#   4013    4283    4401    9552    9625    9790    9793   10584   10902   11450   11451
DVC    = 000200     1843#   4457    4626    4632    4768    4771    4819    4822    4823   10731   10986   11238   11276
                   11279   11412   11680   11721   11724   11919   11924   11927
EARLY   067700     12989#
EBL    = 020000     1778#
ECH    = 000100     1728#   1736   10339   10354   10372   10378   12989
ECI    = 004000     1806#   6020    6042   10374
ECRC   = 001000     1782#
EDT1    074230      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   12989#
EDT110  074246      2714   12989#
EDT111  074250      2721    2728   12989#
EDT114  074252      2742   12989#
EDT2    074240      3191    3198    3394    3401   12989#
EDT223  074254      3240    3429   12989#
EDT336  074264      3016    3774    3788    3795   12989#
EDT337  074274      3781   12989#
EDT344  074304      3816   12989#
EDT353  074316      3865   12989#
ED1     103702     12989#
ED110   103710     12989#
ED111   103714     12989#
ED114   103722     12989#
ED223   103732     12989#
ED336   103742     12989#
ED337   103754     12989#
ED353   103766     12989#
EECC   = 000020     1787#
EFT1    074320      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   12989#
EFT110   074336     2715   12989#
EFT111   074340     2722    2729   12989#
EFT114   074342     2743   12989#
EFT2     074330     3192    3199    3395    3402   12989#
EFT223   074344     3241    3430   12989#
EFT336   074354     3017    3775    3789    3796   12989#
EFT337   074364     3782   12989#
EFT344   074374     3817   12989#
EFT353   074404     3866   12989#
EF110    104060   12989#
EF111    104061   12989#
EF114    104063   12989#
EF336    104066   12989#
EF337    104072   12989#
EHT1     074104     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   12989#
EHT110   074126     2713   12989#
EHT111   074132     2720    2727   12989#
EHT114   074136     2741   12989#
EHT2     074116     3190    3197    3393    3400   12989#
EHT223   074142     3239   12989#
EHT256   074154     3428   12989#
EHT336   074166     3015    3773    3787    3794   12989#
EHT337   074200     3780   12989#
EHT344   074212     3815   12989#
EHT353   074224     3864   12989#
EH1      102635   12989#
EH110    102654   12989#
```

EM111	102663	12989#
EM114	102702	12989#
EM223	102731	12989#
EM256	102757	12989#
EM336	103034	12989#
EM337	103073	12989#
EM344	103232	12989#
EM353	103372	12989#
EMS1	074406	12989#
EMS10	074742	12989#
EMS100	077162	12989#
EMS101	077210	12989#
EMS102	077236	12989#
EMS103	077265	12989#
EMS104	077274	12989#
EMS105	077323	12989#
EMS106	077352	12989#
EMS11	075004	12989#
EMS110	077371	12989#
EMS111	077420	12989#
EMS112	077431	12989#
EMS113	077515	12989#
EMS114	077545	12989#
EMS115	077571	12989#
EMS116	077576	12989#
EMS117	077626	12989#
EMS12	075015	12989#
EMS120	077650	12989#
EMS121	077665	12989#
EMS122	077730	12989#
EMS123	077765	12989#
EMS124	100030	12989#
EMS125	100062	12989#
EMS126	100125	12989#
EMS127	100170	12989#
EMS13	075056	12989#
EMS130	100227	12989#
EMS131	100265	12989#
EMS132	100325	12989#
EMS133	100351	12989#
EMS134	100374	12989#
EMS135	100436	12989#
EMS136	100500	12989#
EMS137	100525	12989#
EMS14	075067	12989#
EMS140	100562	12989#
EMS141	100603	12989#
EMS142	100631	12989#
EMS143	100646	12989#
EMS144	100674	12989#
EMS145	100704	12989#
EMS146	100733	12989#
EMS147	100752	12989#
EMS15	075100	12989#
EMS150	100767	12989#
EMS151	101006	12989#

EMS152	101032	12989#
EMS153	101056	12989#
EMS154	101100	12989#
EMS155	101114	12989#
EMS156	101134	12989#
EMS157	101152	12989#
EMS16	075122	12989#
EMS160	101167	12989#
EMS161	101204	12989#
EMS162	101234	12989#
EMS163	101267	12989#
EMS164	101313	12989#
EMS165	101374	12989#
EMS166	101410	12989#
EMS167	101435	12989#
EMS17	075152	12989#
EMS170	101462	12989#
EMS171	101511	12989#
EMS172	101544	12989#
EMS173	101564	12989#
EMS174	101601	12989#
EMS175	101607	12989#
EMS176	101635	12989#
EMS177	101664	12989#
EMS2	074455	12989#
EMS20	075212	12989#
EMS200	101706	12989#
EMS201	101723	12989#
EMS202	101771	12989#
EMS203	102021	12989#
EMS204	102034	12989#
EMS205	102062	12989#
EMS206	102074	12989#
EMS207	102121	12989#
EMS21	075235	12989#
EMS210	102146	12989#
EMS211	102157	12989#
EMS212	102175	12989#
EMS213	102220	12989#
EMS214	102243	12989#
EMS215	102261	12989#
EMS216	102302	12989#
EMS217	102332	12989#
EMS22	075260	12989#
EMS220	102365	12989#
EMS221	102405	12989#
EMS222	102455	12989#
EMS223	102506	12989#
EMS224	102563	12989#
EMS23	075274	12989#
EMS24	075322	12989#
EMS25	075351	12989#
EMS26	075366	12989#
EMS27	075407	12989#
EMS3	074472	12989#
EMS30	075420	12989#

```
EMS31   075430          12989#
EMS32   075457          12989#
EMS33   075506          12989#
EMS34   075534          12989#
EMS35   075605          12989#
EMS36   075634          12989#
EMS37   075663          12989#
EMS4    074535          12989#
EMS40   075711          12989#
EMS41   075740          12989#
EMS42   075766          12989#
EMS43   076015          12989#
EMS44   076044          12989#
EMS45   076117          12989#
EMS46   076162          12989#
EMS47   076211          12989#
EMS5    074600          12989#
EMS50   076240          12989#
EMS51   076267          12989#
EMS52   076315          12989#
EMS53   076327          12989#
EMS54   076341          12989#
EMS55   076363          12989#
EMS56   076412          12989#
EMS57   076467          12989#
EMS6    074630          12989#
EMS60   076515          12989#
EMS61   076530          12989#
EMS62   076557          12989#
EMS63   076575          12989#
EMS64   076623          12989#
EMS65   076641          12989#
EMS66   076707          12989#
EMS67   076757          12989#
EMS7    074675          12989#
EMS70   077004          12989#
EMS71   077016          12989#
EMS72   077031          12989#
EMS73   077041          12989#
EMS74   077056          12989#
EMS75   077075          12989#
EMS76   077103          12989#
EMS77   077134          12989#
EMTVEC= 000030           1631#   3931*   3932*
EMT1    070012           2210   12989#
EMT10   070062           2259   12989#
EMT100  071060           2656   12989#
EMT101  071076           2663   12989#
EMT102  071114           2670   12989#
EMT103  071124           2677   12989#
EMT104  071136           2684   12989#
EMT105  071154           2691   12989#
EMT106  071164           2698   12989#
EMT107  071174           2705   12989#
EMT11   070066           2266   12989#
EMT110  071214           2712   12989#
```

EMT111	071226	2719	12989#
EMT112	071234	2726	12989#
EMT113	071242	2733	12989#
EMT114	071256	2740	12989#
EMT115	071264	2747	12989#
EMT116	071274	2754	12989#
EMT117	071304	2761	12989#
EMT12	070072	2273	12989#
EMT120	071314	2768	12989#
EMT121	071324	2775	12989#
EMT122	071334	2782	12989#
EMT123	071344	2789	12989#
EMT124	071354	2796	12989#
EMT125	071364	2803	12989#
EMT126	071374	2810	12989#
EMT127	071406	2817	12989#
EMT13	070100	2280	12989#
EMT130	071420	2824	12989#
EMT131	071432	2831	12989#
EMT132	071444	2838	12989#
EMT133	071456	2845	12989#
EMT134	071470	2852	12989#
EMT135	071502	2859	12989#
EMT136	071514	2866	12989#
EMT137	071526	2873	12989#
EMT14	070112	2287	12989#
EMT140	071536	2880	12989#
EMT141	071546	2887	12989#
EMT142	071556	2894	12989#
EMT143	071566	2901	12989#
EMT144	071576	2908	12989#
EMT145	071606	2915	12989#
EMT146	071616	2922	12989#
EMT147	071626	2929	12989#
EMT15	070120	2294	12989#
EMT150	071636	2936	12989#
EMT151	071646	2943	12989#
EMT152	071660	2950	12989#
EMT153	071672	2957	12989#
EMT154	071710	2964	12989#
EMT155	071726	2971	12989#
EMT156	071740	2978	12989#
EMT157	071752	2985	12989#
EMT16	070126	2301	12989#
EMT160	071764	2992	12989#
EMT161	071774	2999	12989#
EMT162	072006	3006	12989#
EMT163	072020	12989#	
EMT164	072030	3021	12989#
EMT165	072036	3028	12989#
EMT166	072044	3035	12989#
EMT167	072052	3042	12989#
EMT17	070136	2308	12989#
EMT170	072060	12989#	
EMT171	072070	3056	12989#
EMT172	072100	3063	12989#

EMT173	072110	3070	12989#
EMT174	072120	3077	12989#
EMT175	072132	3084	12989#
EMT176	072140	3091	12989#
EMT177	072146	3098	12989#
EMT2	070016	2217	12989#
EMT20	070146	2315	12989#
EMT200	072160	3105	12989#
EMT201	072172	3112	12989#
EMT202	072204	3119	12989#
EMT203	072216	3126	12989#
EMT204	072230	3133	12989#
EMT205	072246	3140	12989#
EMT206	072256	3147	12989#
EMT207	072266	3154	12989#
EMT21	070156	2322	12989#
EMT210	072300	3161	12989#
EMT211	072306	3168	12989#
EMT212	072322	3175	12989#
EMT213	072336	3182	12989#
EMT214	072346	3189	12989#
EMT215	072366	3196	12989#
EMT216	072400	3203	12989#
EMT217	072410	3210	12989#
EMT22	070166	2329	12989#
EMT220	072420	3217	12989#
EMT221	072430	3224	12989#
EMT222	072440	3231	12989#
EMT223	072450	3238	12989#
EMT224	072460	12989#	
EMT225	072462	12989#	
EMT226	072464	12989#	
EMT227	072466	12989#	
EMT23	070176	2336	12989#
EMT230	072470	12989#	
EMT231	072472	12989#	
EMT232	072474	12989#	
EMT233	072476	12989#	
EMT234	072500	12989#	
EMT235	072502	12989#	
EMT236	072504	12989#	
EMT237	072506	12989#	
EMT24	070206	2343	12989#
EMT240	072510	12989#	
EMT241	072512	12989#	
EMT242	072514	12989#	
EMT243	072516	12989#	
EMT244	072520	12989#	
EMT245	072522	12989#	
EMT246	072524	3371	12989#
EMT247	072534	3378	12989#
EMT25	070216	2350	12989#
EMT250	072546	3385	12989#
EMT251	072562	3392	12989#
EMT252	072570	3399	12989#
EMT253	072576	3406	12989#

EMT254	072614	3413	12989#
EMT255	072624	3420	12989#
EMT256	072634	3427	12989#
EMT257	072644	3434	12989#
EMT26	070226	2357	12989#
EMT260	072656	3441	12989#
EMT261	072670	3448	12989#
EMT262	072710	3455	12989#
EMT263	072720	3462	12989#
EMT264	072730	3469	12989#
EMT265	072750	3476	12989#
EMT266	072770	3483	12989#
EMT267	073002	3491	12989#
EMT27	070236	2364	12989#
EMT270	073020	3498	12989#
EMT271	073034	3506	12989#
EMT272	073052	3513	12989#
EMT273	073070	3520	12989#
EMT274	073106	3528	12989#
EMT275	073124	3536	12989#
EMT276	073136	3544	12989#
EMT277	073152	3553	12989#
EMT3	070024	2224	12989#
EMT30	070246	2371	12989#
EMT300	073170	3561	12989#
EMT301	073210	3569	12989#
EMT302	073232	3576	12989#
EMT303	073244	3583	12989#
EMT304	073256	3590	12989#
EMT305	073270	3597	12989#
EMT306	073302	3604	12989#
EMT307	073312	3611	12989#
EMT31	070256	2378	12989#
EMT310	073332	3618	12989#
EMT311	073344	3625	12989#
EMT312	073356	3632	12989#
EMT313	073370	3639	12989#
EMT314	073410	3646	12989#
EMT315	073422	3653	12989#
EMT316	073434	3660	12989#
EMT317	073446	3667	12989#
EMT32	070266	2385	12989#
EMT320	073456	3674	12989#
EMT321	073466	3681	12989#
EMT322	073476	3688	12989#
EMT323	073504	3695	12989#
EMT324	073514	3702	12989#
EMT325	073526	3709	12989#
EMT326	073540	3716	12989#
EMT327	073556	3723	12989#
EMT33	070276	2392	12989#
EMT330	073570	3730	12989#
EMT331	073600	3737	12989#
EMT332	073612	3744	12989#
EMT333	073624	3751	12989#
EMT334	073642	3758	12989#

EMT335	073660	3765	12989#
EMT336	073700	3014	3772 12989#
EMT337	073710	3779	12989#
EMT34	070306	2399	12989#
EMT340	073720	3786	12989#
EMT341	073732	3793	12989#
EMT342	073740	3800	12989#
EMT343	073752	3807	12989#
EMT344	073764	3814	12989#
EMT345	073776	3821	12989#
EMT346	074006	3828	12989#
EMT347	074022	3835	12989#
EMT35	070316	2406	12989#
EMT350	074034	3842	12989#
EMT351	074052	3849	12989#
EMT352	074062	3856	12989#
EMT353	074066	3863	12989#
EMT354	074076	3870	12989#
EMT36	070326	2413	12989#
EMT37	070336	2420	12989#
EMT4	070032	2231	12989#
EMT40	070346	2427	12989#
EMT41	070354	2434	12989#
EMT42	070364	2441	12989#
EMT43	070376	2448	12989#
EMT44	070406	2455	12989#
EMT45	070416	2462	12989#
EMT46	070426	2469	12989#
EMT47	070436	2476	12989#
EMT5	070040	2238	12989#
EMT50	070444	2483	12989#
EMT51	070454	2490	12989#
EMT52	070466	2497	12989#
EMT53	070504	2505	12989#
EMT54	070522	2512	12989#
EMT55	070532	2519	12989#
EMT56	070544	2526	12989#
EMT57	070562	2533	12989#
EMT6	070046	2245	12989#
EMT60	070572	2541	12989#
EMT61	070610	2549	12989#
EMT62	070630	2557	12989#
EMT63	070644	12989#	
EMT64	070646	2571	12989#
EMT65	070666	2578	12989#
EMT66	070706	2585	12989#
EMT67	070720	2592	12989#
EMT7	070054	2252	12989#
EMT70	070732	2599	12989#
EMT71	070742	2606	12989#
EMT72	070752	2613	12989#
EMT73	070770	2621	12989#
EMT74	071006	2628	12989#
EMT75	071016	2635	12989#
EMT76	071036	2642	12989#
EMT77	071046	2649	12989#

```
ENRGDT  070012          12989#
EQUALS  066307          12989#
ERR     = 040000         1706#  4620   6249   6358   6473   6588   6712   9971   9973   9989  10023
ERRNMB  043152          9162#  9163#  9166   9178   9183   9275#
ERRTYP  042414          9141#  12518
ERRVEC= 000004          1624#  3945   3946#  3957#  4224   4225   4226#  4227#  4238#  4239#  4243#  4244#  4270
                        4271   4272#  4273#  4277#  4278#  4289#  4290#  6974   6975   6976#  6977#  7050#  7051#
                        8155   8156   8157#  8158#  8236#  8237#  9543   9544   9548#  9549#  9580#  9581#  9616
                        9617   9621#  9622#  9647#  9648#  9659   9660   9661#  9662#  9668#  9676#  9677#  9691
                        9692   9693#  9694#  9718#  9719#  10566  10567  10570#  10571#  10604#  10605#  10852  10853
                       10854#  10855#  10866#  10867#  12378  12379#  12381#  12384#
ERTY00  043160          9148   9280#
ERTY01  043166          9156   9281#
ERTY02  043176          9165   9283#
ERTY03  043205          9171   9285#
ERTY04  043213          9267   9287#
ESRC    = 004000         1780#
FER     = 000020         1730#  1736   10303
FIND    = 000001         4531#  4532   4539#  4540   4693#  4694   4700#  4701   4905#  4906   4912#  4913   4920#
                        4921   4942#  4944#  4965#  4966   4972#  4973   4979#  4980   5001#  5003#  5041#  5042
                        5048#  5049   5055#  5056   5076#  5078#  5117#  5118   5127#  5128   5148#  5150#  5176#
                        5177   5183#  5184   5190#  5191   5212#  5214#  5237#  5239#  5241#  5242   5247#  5248
                        5254#  5255   5261#  5262   5278#  5280#  5309#  5310   5316#  5317   5323#  5324   5341#
                        5343#  5363#  5364   5370#  5373#  5379#  5380   5386#  5387   5404#  5406#  5427#  5428
                        5446#  5449#  5455#  5456   5462#  5463   5480#  5482#  5508#  5511#  5532#  5533   5545#
                        5548#  5554#  5555   5561#  5562   5579#  5581#  5624#  5625   5637#  5640#  5657#  5659#
                        5715#  5716   5722#  5723   5740#  5742#  5795#  5796   5814#  5815   5821#  5824#  5830#
                        5831   5837#  5838   5855#  5857#  5928#  5930#  5951#  5952   5970#  5973#  5979#  5980
                        5986#  5987   6005#  6007#  6035#  6036   6057#  6060#  6066#  6067   6073#  6074   6091#
                        6093#  6113#  6114   6160#  6162#  6234#  6235   6275#  6277#  6343#  6344   6384#  6386#
                        6458#  6459   6499#  6501#  6573#  6574   6614#  6616#  6697#  6698   6738#  6740#  6745#
                        6748#  6749   6778#  6779   6784#  6785#  6786   6791#  6792   6814#  6816#  6821#  6850#
                        6851   6884#  6885   6908#  6910#  6917#  6918   6933#  6934   6939#  6940   6966#  6968#
                        6988#  6989   7000#  7001   7068#  7070#  7096#  7097   7103#  7104   7110#  7113#  7119#
                        7120   7141#  7143#  7169#  7170   7176#  7177   7183#  7186#  7192#  7193   7215#  7217#
                        7241#  7242   7248#  7249   7255#  7258#  7264#  7265   7294#  7296#  7324#  7325   7331#
                        7332   7338#  7341#  7347#  7348   7374#  7376#  7404#  7405   7411#  7412   7418#  7421#
                        7427#  7428   7455#  7457#  7484#  7485   7491#  7492   7498#  7501#  7507#  7508   7535#
                        7537#  7564#  7565   7571#  7572   7578#  7581#  7587#  7588   7611#  7613#  7641#  7642
                        7654#  7655   7660#  7661   7667#  7668   7675#  7678#  7684#  7685   7706#  7708#  7736#
                        7737   7755#  7756   7762#  7763   7770#  7773#  7779#  7780   7805#  7807#  7835#  7836
                        7854#  7855   7861#  7862   7869#  7872#  7878#  7879   7942#  7943   7956#  7959#  7965#
                        7966   7983#  7985#  8024#  8025   8031#  8032   8038#  8041#  8047#  8048   8069#  8071#
                        8101#  8109#  8110   8116#  8117   8123#  8126#  8132#  8133   8184#  8185   8249#  8251#
                        8279#  8280   8286#  8287   8306#  8307   8313#  8314   8320#  8323#  8329#  8330   8358#
                        8360#  8390#  8391   8397#  8398   8417#  8418   8424#  8425   8431#  8434#  8440#  8441
                        8472#  8474#  8502#  8503   8515#  8516   8522#  8525#  8531#  8532   8567#  8569#  8598#
                        8599   8612#  8615#  8621#  8622   8655#  8657#  8685#  8686   8698#  8699   8705#  8708#
                        8714#  8715   8749#  8751#  8789#  8790   8803#  8806#  8812#  8813   8829#  8831#  8870#
                        8871   8877#  8878   8884#  8887#  8893#  8894   8910#  8912#  8949#  8950   8956#  8957
                        8963#  8966#  8972#  8973   9011#  9013#  9039#  9040   9046#  9047   9053#  9056#  9062#
                        9063
FMT1B = 010000          1805#  6020   6042   6213   6322   6437   6552   6676   6769   7787   7789   7886   7888
                        8160   8212   8229   8338   8340   8452   8457   8629   8631   8722   8724   8918  10663
                       11654
FNCDTB  067254          6240   6248   6349   6357   6464   6472   6579   6587   6703   6711  10017  12989#
```

```
ENCMSK= 000077          1648#  11450
FD    = 000002          1646#   9916
F1    = 000004          1645#   9916
F2    = 000010          1644#   9916  10490
F3    = 000020          1643#   9916
F4    = 000040          1642#   9916
GET     044216          4317    4393   4525   4575   4612   4687   4733   4784   4841   4899   4959   5026   5035
                        5111    5170   5241   5303   5357   5421   5496   5526   5595   5618   5673   5703   5789
                        5808    5910   5945   6029   6107   6136   6228   6337   6452   6567   6691   6785   6850
                        6884    6939   6994   7090   7163   7235   7318   7398   7478   7558   7635   7654   7730
                        7749    7829   7848   7936   8006   8018   8103   8178   8273   8300   8384   8411   8496
                        8592    8679   8783   8852   8864   8943   9033   9359   9383   9410   9433   9459   9534#
GETBUF  001326          2112#   9546
GETINX  001504          2183#   4314*  4315*  4572*  4573*  4574*  9381*  9382*  9431*  9432*  9492   9547
GETSTS  044132          4392    4524   4611   4685   4732   4897   4957   5025   5109   5168   5237   5301   5356
                        5419    5495   5594   5672   5781   5879   5943   6027   6106   6166   6281   6390   6505
                        6620    6745   6821   6973   7088   7161   7233   7316   7396   7476   7556   7627   7722
                        7821    7929   7999   8089   8170   8266   8377   8488   8584   8671   8776   8845   8931
                        9031    9358   9409   9452   9488#
GNS   = ****** U        1955    3983   9100   9107  12873  12874  12875  12876  12877  12878  12880  12882  12883
                       12884   12885  12886  12887
GO    = 000001          1647#   4679   4834   4891   4951   5019   5103   5157   5221   5230   5286   5350   5413
                        5489    5519   5588   5666   5696   5775   5882   5937   6021   6100   6200   6207   6316
                        6424    6431   6539   6546   6654   6670   6763   6843   6868   6877   7077   7150   7222
                        7304    7383   7464   7544   7617   7712   7811   7919   7992   8078   8092   8163   8259
                        8292    8369   8404   8481   8577   8664   8768   8836   8919   8935   9020   9393   9442
                        9704    9815   9827   9832   9848   9938   9941
GTSWR = 104407          3978    4199  12880#
HCE   = 000200          1727#   1736  10266  10271  10286  10288  10301  10316  12989
HCI   = 002000          1807#   6020   6042  10264
HCRC  = 000400          1726#   1736  10273
HELP    106126          4060   12989#
HELPQS  066317          4054   12989#
HT    = 000011          1534#   9204  12315  12356
IAE   = 002000          1724#   6465   6467   7676   7771   7870   8508   8510   8512   8523   8604   8606   8608
                        8613    8691   8693   8695   8706  10170  10467  10482  10508  10654  10671  11170  11224
                       11227   11645  11663  12989
IDXMSK= 000077          1883#   9560   9563   9633   9636
IE    = 000100          1897#   1930#  5775   5882
ILF   = 000001          1734#   6241   6243  10042  10044  11064  11124  11128  11170  11191  11194  12989
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*  6260   6369   6484   6599   6723   6800
ILR   = 000002          1733*  7008   7036  11064  11110  11114  11378  11382  11385
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*  4414  10927
IVC   = 010000          1840*  5123   6350   6352   7948   7950   7952   7958   8795   8797   8799   8805  10085
                       10086  10689  10812  11238  11245  11248  11317  11680  11704  11707  11794  11888  12989
LBC   = 002000          1842*  4846   4849   4850
LBT   = 002000          1710*  4620  10245  10469  10471
LF    = 000012          1535*  9202  12059  12350  12356  12989
LS    = 000004          1789*  4437  10951  11503
LSC   = 004000          1841*  5289
LST   = 000002          1790*  4437  10951  11503
MCLK  = 004000          1762*
MCPE  = 020000          1891*  4582   9815   9832   9845   9848  11625  11630
MDF   = 000100          1767*  4777
MDPE  = 000400          1908*  4587  10341
MEDENB  001472          2171*  4210*  9392*
MFGFIL  104106         12989*
MI    = 000004          1770*
MIXED   067364         12989*
MOC   = 000400          1765*  4777   6189   6304   6413   6528   6643   6663   6834   6859
MOH   = 020000          1797*
MOL   = 010000          1708*  4617   4621   4648   4744   4747   4795   4798   4799  10750  10760  10761  10766
                       10771  10799  11050  11053  11214  11291  11292  11299  11304  11463  11744  11755  11756
                       11760  11763  11865  11866  11870  11875
MRD   = 002000          1763*
MS    = 000040          1768*
MSC   = 000002          1771*
MSE   = 100000          1848*
MSER  = 000200          1766*  4777
MUR   = 001000          1764*  4777   6189   6304   6413   6528   6643   6834   6859
MWD   = 000010          1788*  4438  10952  11504
MWP   = 000010          1769*  4777   6528
MXF   = 001000          1907*  4587  10230
NDTMSK= 115760          1736*  5371   5447   5546   5822   5971   6058   7111   7184   7256   7339   7419   7499
                        7579  7676   7771   7870   7957   8039   8124   8321   8432   8523   8613   8706   8804
                        8885  8964   9054
NED   = 010000          1904*  4010   4279   4587   9797   9801  10579
NEM   = 004000          1905* 10215
NOP   = 000000          1652*  4538   5350   5519   5696   5894   6167   6282   6391   6506   6621   6746   6868
```

```
NOTEX   067230          4018  12989#
NSA   = 100000          1795#
OCC   = 100000          1776#
OFD   = 000200          1808#  6020  6042
OFFSET= 000014          1658#  5221  5413  5489  5588  5666  5775  5882  6100  6877  8078  8919
OM    = 000001          1715#  4648  5433  5436  5437  5957  5961  6119  6122  6123  6141  6144  11291
                       11347  11350  11462
ONES    067424         12989#
OPE   = 020000          1839#
OPI   = 020000          1721#  6704  6706  10069  10744  10768  11064  11082  11086  11170  11207  11210  11301
                       11737  11740  11765  11872  12989
OR    = 000200          1910#
PACACK= 000022          1662#
PAKACK= 000022          1661#  1662  4679  4891  6200  6424  6539  6654  6843  9393
PAR   = 000010          1731#  6945  6947  9860  9864  9876  10636  11170  11175  11180  11605  11612
PAT   = 000020          1913#  6925
PDA   = 000400          1783#
PGE   = 002000          1906#  4587
PGM   = 001000          1711#  4648  11462
PHA   = 000200          1784#
PIP   = 020000          1707#  4620  4750  4753  4754  4801  4804  5031  8011  8014  8857  8861  9440
                       10760  10781  10786  11291  11362  11367  11462  11755  11776  11781  11865  11902  11907
PIRQ  = 177772          1541#
PIRQVE= 000240          1635#
PLFS  = 002000          1781#
PRIERR  045212          4693  4905  4965  5041  5117  5176  5247  5309  5363  5427  5532  5624  5795
                        5814  5951  6035  6113  6234  6343  6458  6573  6697  6791  7000  7096  7169
                        7241  7324  7404  7484  7564  7660  7755  7854  7942  8024  8109  8184  8279
                        8306  8390  8417  8502  8598  8685  8789  8870  8949  9039  9763#
PROMPT  066311          4135  4144  12989#
PR0   = 000000          1558#
PR1   = 000040          1559#
PR2   = 000100          1560#
PR3   = 000140          1561#  12038
PR4   = 000200          1562#
PR5   = 000240          1563#
PR6   = 000300          1564#  3916  4205  4227  4273  5752  5761  5866  5869  6977  8158  9549  9622
                        9662  9694  10571  10855  12044
PR7   = 000340          1565#  5759  5875
PS    = 177776          1538#  1539
PSEL  = 002000          1893#
PSW   = 177776          1539#
PUT     044466          4362  4384  4491  4512  4567  4680  4727  4778  4835  4874  4883  4892  4952
                        5011  5020  5086  5095  5104  5163  5222  5231  5296  5351  5414  5490  5520
                        5589  5612  5667  5688  5697  5768  5776  5883  5895  5938  6022  6101  6130
                        6181  6190  6201  6222  6296  6305  6331  6405  6414  6425  6446  6520  6529
                        6540  6561  6635  6644  6655  6664  6685  6757  6778  6826  6835  6844  6860
                        6869  6878  6933  7083  7156  7228  7311  7390  7471  7551  7648  7743  7842
                        7913  7930  8000  8084  8095  8172  8267  8293  8378  8405  8490  8586  8673
                        8759  8777  8846  8926  8936  9026  9396  9445  9610#
PUTBUF  001376          2139#  9619
PUTINX  001533          2188#  4359*  4360*  4376  4488*  4489*  4504  4564*  4565*  4677*  4678*  4724*  4725*
                        4775*  4776*  4832*  4833*  4871*  4872*  4880*  4881*  4889*  4890*  4949*  4950*  5008*
                        5009*  5017*  5018*  5083*  5084*  5092*  5093*  5101*  5102*  5158  5219*  5220*  5228*
                        5229*  5290  5348*  5349*  5411*  5412*  5487*  5488*  5517*  5518*  5586*  5587*  5610*
                        5611*  5664*  5665*  5685*  5686*  5694*  5695*  5766*  5767*  5773*  5774*  5880*  5881*
```

```
                    5892*  5893*  5935*  5936*  6012   6098*  6099*  6128*  6129*  6178*  6179*  6187*  6188*
                    6198*  6199*  6214   6293*  6294*  6302*  6303*  6323   6402*  6403*  6411*  6412*  6422*
                    6423*  6438   6517*  6518*  6526*  6527*  6537*  6538*  6553   6632*  6633*  6641*  6642*
                    6652*  6653*  6661*  6662*  6677   6754*  6755*  6770   6823*  6824*  6832*  6833*  6841*
                    6842*  6857*  6858*  6866*  6867*  6875*  6876*  6928   7078   7151   7223   7305   7384
                    7465   7545   7621   7716   7815   7910*  7911*  7923   7993   8079   8093*  8094*  8164
                    8260   8371   8482   8578   8665   8756*  8757*  8769   8839   8920   8933*  8934*  9021
                    9394*  9395*  9443*  9444*  9620   9867
PWRVEC= 000024       1630*  3935*  3936*  12892* 12893* 12902* 12908* 12920* 12921*
QSTMRK  066315       4049   4154   12989*
RCLSTS  055152       4972   5183   5254   9466   11161*
RD    = 000070       1682*
RDCHR = 104411       4041   4055   4065   4068   4136   4145   12745  12883*
RDLIN = 104412      12815  12884*
RDOCT = 104413       4079   4095   4116   12885*
RDY   = 000200       1896*  4401   9704   9705   9811   9814   10902
READY   007106       4046   4203*  9075   9123
RECAL = 000006       1655*  4951   5019   5103   5157   5230   9442
RESREG= 104415       9271   12887*
RESVEC= 000010       1625*
REX   = 010000       1779*
RG    = 040000       1777*
RGDTPT  067364      12989*
RH    = 000072       1683*
RIP   = 000020       1660*  6021
RLEASE= 000012       1657*
RMAS  = 000016       1857*  5610   5766
RMASI   001344       2122*  4426   11483  12989
RMASO   001414       2149*  5609*  5765*
RMBA  = 000004       1939*  4232*  4233   4377   4505   6215   6324   6439   6554   6678   6771
RMBAE = 000050       1942*
RMBAI   001332       2117*  4407   10417  10419  10913  12989
RMBAO   001402       2144*  4369*  4498*  6209*  6318*  6433*  6548*  6672*  6765*  10413  10416
RMCC  = 000036       1864*
RMCCI   001364       2130*
RMCCO   001434       2157*
RMCS1 = 000000       1853*  1937*  4012   4229*  4275   4381   4509   4564   4572   4677   4832   4889   4949
                     5017   5101   5161   5219   5228   5294   5348   5411   5487   5517   5586   5664   5694
                     5773   5880   5892   5935   6016   6098   6198   6220   6329   6422   6444   6537   6559
                     6652   6683   6776   6841   6866   6875   7037*  7081   7154   7226   7308   7387   7468
                     7548   7625   7720   7819   7927   7997   8082   8093   8168   8264   8375   8486   8582
                     8669   8774   8843   8924   8933   9024   9394   9443   9551   9624   9703   10577
RMCS1I  001326       2115*  4399   4580   9790   9792   9794   9811   9813   9816   9827   9831   9833   9845
                     9847   9849   9938   9940   9951   9994   10400  10900  11449  11625  11629  11631  12989
RMCS10  001376       2142*  4373*  4502*  4538*  4566*  4679*  4834*  4891*  4951*  5019*  5103*  5157*  5221*
                     5230*  5286*  5350*  5413*  5489*  5519*  5588*  5666*  5696*  5775*  5882*  5894*  5937*
                     6021*  6100*  6200*  6207*  6208*  6316*  6317*  6424*  6431*  6432*  6539*  6546*  6547*
                     6654*  6670*  6671*  6763*  6764*  6843*  6868*  6877*  7077*  7150*  7222*  7304*  7383*
                     7464*  7544*  7617*  7712*  7811*  7919*  7992*  8078*  8092*  8163*  8259*  8292*  8369*
                     8404*  8481*  8577*  8664*  8768*  8836*  8919*  8935*  9020*  9393*  9442*  9915   10431
RMCS2 = 000010       1940*  4007*  4008*  4009   4234*  4235   4276   4382   4391*  4510   4573   6929   9550
                     9623   10576* 10578  10857* 10859*
RMCS2I  001336       2119*  4411   4586   9772   9797   9799   9800   9986   10057  10158  10201  10216  10231
                    10342  10414  10922  12989
RMCS20  001406       2146*  4374*  4503*  6926*
RMCS3 = 000052       1943*
```

```
RMDA   = 000006  1854#  5160   5291   6013   6218   6327   6442   6557   6681   6774   6930   7080   7153
                 7225   7306   7385   7467   7546   7623   7719   7816   7925   7994   8081   8165   8261
                 8374   8485   8579   8668   8772   8841   8921   9023   11656
RMDAI    001334  2118#  6046   6048   10513  12989
RMDAO    001404  2145#  5156*  5285*  6018*  6211*  6320*  6435*  6550*  6674*  6767*  6927*  7075*  7148*
                 7221*  7303*  7382*  7463*  7543*  7620*  7689*  7690   7714*  7784*  7785   7790*  7812*
                 7921*  7990*  8076*  8161*  8256*  8335*  8336   8342*  8364*  8448*  8449   8458*  8479*
                 8536*  8537   8575*  8626*  8627   8632*  8662*  8766*  8838*  8916*  9018*  10439  10440
                10659  10661  10665  11650  11652
RMDB   = 000022  1941#  4236*  4237   4359   4488   9499
RMDBI    001350  2124#
RMDBO    001420  2151#  4358*  4487*
RMDC   = 000034  1863#  5159   6014   6128   6219   6328   6443   6558   6682   6775   7079   7152   7224
                 7307   7386   7466   7547   7624   7718   7817   7926   7995   8080   8166   8262   8373
                 8483   8580   8667   8771   8840   8922   9022
RMDCI    001362  2129#  6051   6053   10525  12989
RMDCO    001432  2156#  5155*  6019*  6127*  6212*  6321*  6436*  6551*  6675*  6768*  7076*  7149*  7213*
                 7269   7271*  7302*  7381*  7434   7436*  7462*  7514   7516*  7542*  7594   7596*  7619*
                 7713*  7813*  7883*  7884   7889*  7922*  7991*  8077*  8162*  8257*  8370*  8403*  8480*
                 8576*  8663*  8719*  8720   8725*  8765*  8837*  8917*  9019*  10441  10656  11647
RMDS   = 000012  1855#  9381   9431
RMDSI    001340  2120#  4460   4617   4619   4622   4645   4647   4649   4744   4746   4750   4752   4756
                 4758   4795   4797   4801   4803   4807   4809   4825   4827   5031   5433   5435   5439
                 5441   5501   5504   5538   5540   5600   5602   5630   5632   5678   5680   5708   5711
                 5957   5960   5963   5965   6119   6121   6141   6143   6249   6253   6358   6362   6473
                 6477   6588   6592   6712   6716   8011   8013   8857   8860   9390   9440   9829   9925
                 9927   9972   9989   10023  10027  10083  10110  10245  10470  10695  10750  10759  10766
                10770  10772  10781  10785  10787  10796  10798  10800  10810  10814  10816  10995  11036
                11038  11040  11050  11052  11054  11214  11252  11290  11299  11303  11305  11315  11319
                11320  11331  11333  11335  11347  11349  11351  11362  11366  11368  11461  11711  11744
                11754  11760  11762  11764  11776  11780  11782  11792  11796  11798  11808  11810  11812
                11864  11870  11874  11876  11886  11890  11891*  11892  11902  11906  11908  12989
RMDSO    001410  2147#
RMDT   = 000026  1860#  4314
RMDTI    001354  2126#  4316*  4321   4323   4325   4327   12989
RMDTO    001424  2153#
RMEC1  = 000044  1867#
RMEC1I   001372  2133#  12989
RMEC1O   001442  2160#
RMEC2  = 000046  1868#  9494
RMEC2I   001374  2134#  4444   9493   10963  11527  12989
RMEC2O   001444  2161#
RMER1  = 000014  1856#  4378   4506   5008   5293   5685   6754   7996   8842   9868
RMER1I   001342  2121#  4421   4634   4636   4640   4642   6242   6466   6581   6705   6891   6894   6945
                 6948   7007   8508   8511   8604   8607   8691   8694   9860   9875   9877   9967   10043
                10070  10113  10128  10171  10248  10274  10289  10304  10357  10376  10379  10467  10482
                10497  10508  10636  10643  10670  10744  10747  10768  10940  11064  11068  11070  11071
                11082  11084  11085  11096  11098  11099  11110  11112  11113  11124  11126  11127  11170
                11175  11179  11181  11191  11193  11195  11207  11209  11211  11224  11226  11228  11301
                11378  11382  11384  11386  11396  11398  11400  11410  11414  11416  11474  11605  11611
                11613  11662  11737  11739  11741  11765  11872  11983  12989
RMER1O   001412  2148#  4370*  4499*  5010*  5287*  5687*  6756*  7989*  8835*  9864
RMER2  = 000042  1866#  4379   4507   5292
RMER2I   001370  2132#  4456   4626   4628   4632   4653   4655   4762   4764   4768   4770   4813   4815
                 4819   4821   4846   4848   5123   6351   7948   7951   8795   8798   9862   9969   10087
                10143  10319  10638  10689  10692  10703  10731  10734  10783  10812  10985  11177  11238
```

```
                    11245   11247   11249   11262   11264   11266   11276   11278   11280   11317   11364   11412   11536
                    11607   11680   11684   11704   11706   11708   11721   11723   11725   11778   11794   11888   11904
                    11918   11924   11926   11928   11938   11940   11942   11986   12989
RMER20  001440       2159#   4371#   4500#   5288#   5289#
RMLA  = 000020       1858#   8193    8194    8203
RMLAI   001346       2123#  12989
RMLAO   001416       2150#
RMMR1 = 000024       1859#   4380    4508    4724    4775    4871    4880    5083    5092    6178    6187    6293    6302
                     6402    6411    6517    6526    6632    6641    6661    6823    6832    6857    7910    7924    8756
                     8770
RMMR1I  001352       2125#   4436    4738    4741    4789    4792   10950   11502   12989
RMMR1O  001422       2152#   4372#   4501#   4726#   4777#   4873#   4882#   5085#   5094#   6180#   6189#   6295#   6304#
                     6404#   6413#   6519#   6528#   6634#   6643#   6663#   6825#   6834#   6859#   7912#   7920#   8758#
                     8764#
RMMR2 = 000040       1865#
RMMR2I  001366       2131#   4448   10972   11514   12989
RMMR2O  001436       2158#
RMOF  = 000032       1862#   6015    6217    6326    6441    6556    6680    6773    7622    7717    7818    8167    8263
                     8372    8484    8581    8666    8773    8923
RMOFI   001360       2128#   6041   10264   10374   11654   12989
RMOFO   001430       2155#   6020#   6213#   6322#   6437#   6552#   6676#   6769#   7618#   7715#   7787    7789#   7814#
                     7886    7888#   8160#   8212    8229    8231#   8258#   8338    8340#   8365#   8452    8457#   8478#
                     8574#   8629    8631#   8661#   8722    8724#   8767#   8918#  10663   12989
RMR   = 000004       1732#   6891    6893   11064   11096   11100   11378   11396   11399
RMSN  = 000030       1861#
RMSNI   001356       2127#  12989
RMSNO   001426       2154#
RMWC  = 000002       1938#   4230#   4231    6216    6325    6440    6555    6679    6772
RMWCI   001330       2116#  10398   10410   10428   10492   12989
RMWCO   001400       2143#   6210#   6319#   6434#   6549#   6673#   6766#  10411   10429
RQA   = 100000       1817#   4449   10973   11515
RQB   = 040000       1818#   4449   10973   11515
RTC   = 000016       1659#   5937
SADMSK= 000037       1701#
SAVREG= 104414       9142   12886#
SA1   = 000001       1696#
SA16  = 000020       1692#
SA2   = 000002       1695#
SA4   = 000004       1694#
SA8   = 000010       1693#
SC    = 100000       1889#   4400    9815    9832    9848   10901
SCHSTS  057516       8313    8424    8956   11596#
SCTMSG  066176      12989#
SCTMSK= 003700       1752#
SCO   = 000100       1750#
SC1   = 000200       1749#
SC2   = 000400       1748#
SC3   = 001000       1747#
SC4   = 002000       1746#
SEARCH= 000030       1665#   8163    8292    8404    8481    8577    8664    8768    8836    8935    9991   10392
SECERR  046044       4539    4700    4920    4979    5055    5127    5190    5261    5323    5386    5462    5561    5722
                     5837    5986    6073    7119    7192    7264    7347    7427    7507    7587    7684    7779    7878
                     7965    8047    8132    8329    8440    8531    8621    8714    8812    8893    8972    9062    9911#
SEEK  = 000004       1654#   7077    7150    7222    7304    7383    7464    7544    7617    7712    7811    7919    7992
                     8092    8259    8369    9020
SEKSTS  052120       7103    7176    7248    7331    7411    7491    7571    7667    7762    7861    8116    8286    8397
```

```
                    9046  10625#
SHUT    062030     12052#  12586  12669
SIZCLK  044704      4186   9658#
SKI   = 040000      1838#  4457   4653   4762   4765   4813   4816   4817   10655  10658  10674  10704  10706
                   10710  10721  10783  10986  11238  11262  11265  11364  11646  11649  11666  11680  11685
                   11687  11692  11778  11904  11919  11938  11941
SNGPRT= 020024      1800#  4321   4325   4329
STACK = 001100      1529#  3927   4219   4259   4303   4346   4476   4553   4600   4666   4713   4860   4938
                    4996   5072   5144   5207   5274   5337   5400   5475   5574   5652   5735   5850   5924
                    6000   6086   6155   6270   6379   6494   6609   6733   6809   6903   6961   7063   7136
                    7209   7289   7369   7450   7530   7605   7701   7800   7899   7978   8064   8144   8245
                    8354   8467   8563   8650   8744   8824   8906   8994
STANDA  006126      3996   4031#
START   005322      1975   3911#  4248   12058
STCDRV  061062      5048   5379   5455   5554   5715   5830   5979   6066   8031   8515   8698   8877   11857#
STKLMT= 177774      1540#
STOP    062000     12035#  12372
STSD1   103776     12989#
STSD2   104014     12989#
STSD3   104034     12989#
STSD4   104046     12989#
STSF    104077     12989#
STSH1   103441     12989#
STSH2   103516     12989#
STSH3   103605     12989#
STSH4   103643     12989#
SWR     001154      2028#  3925   3947*  3949   3955*  3962*  3976   4197   9143   12373  12387  12390  12405
                   12412  12509  12516  12528  12532  12589  12628  12683*  12900  12913*
SWREG   000176      1957#  3955   3976   4197   12589  12628  12651
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#  9143
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#
TADMSK= 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  045026  4686  4898  4958  5034  5110  5169  5239  5302  5420  5780  5887  5944  6028
                6227  6336  6451  6566  6690  6784  7089  7162  7234  7317  7397  7477  7557
                7653  7748  7847  8017  8090  8101  8177  8272  8299  8383  8410  8495  8591
                8678  8782  8863  8932  8942  9032  9453  9687#
TKVEC  = 000060  1633#  12566*  12567*
TPVEC  = 000064  1634#
TRAPVE= 000034  1632#  3933*  3934*
TRE    = 040000  1890#  4566  4582  9815  9832  9848  9993  9995  10400
TRTVEC= 000014  1627#
TST    = 010000  1820#  4450  10974  11516
TSTNMB  043150  9154*  9155*  9157  9274#
TSTPRP  043216  4942  5001  5076  5148  5212  5278  5341  5404  5480  5579  5657  5740  5855
                5928  6005  6091  6160  6275  6384  6499  6614  6738  6814  6908  6966  7068
                7141  7215  7294  7374  7455  7535  7611  7706  7805  7983  8069  8249  8358
                8472  8567  8655  8749  8829  8910  9011  9322#
TSTQUE  001446  2167#  4169  4170*  4209  4221  4261  4305  4348  4478  4555  4602  4668  4715
                4862  4940  4998  5074  5146  5209  5276  5339  5402  5477  5576  5654  5737
                5852  5926  6002  6088  6157  6272  6381  6496  6611  6735  6811  6905  6963
                7065  7138  7211  7291  7371  7452  7532  7607  7703  7802  7901  7980  8066
                8146  8247  8356  8469  8565  8652  8746  8826  8908  8996  9070  9072*  9076*
                10573  10858  10925  11486
TST1    007140  4214#  12432
TST10   011656  4663#  12439
TST11   012036  4710#  12440
TST12   013110  4857#  12441
TST13   013410  4931#  12442
TST14   013642  4989#  12443
TST15   014164  5065#  12444
TST16   014510  5137#  12445
TST17   014766  5200#  12446
TST2    007336  4256#  12433
TST20   015260  5271#  12447
TST21   015530  5334#  12448
TST22   015760  5397#  12449
TST23   016304  5472#  12450
TST24   016724  5571#  12451
TST25   017270  5649#  12452
TST26   017662  5732#  12453
TST27   020336  5847#  12454
TST3    007522  4300#  12434
TST30   020670  5921#  12455
TST31   021212  5997#  12456
TST32   021574  6083#  12457
TST33   022112  6152#  12458
TST34   022676  6267#  12459
TST35   023422  6376#  12460
TST36   024212  6491#  12461
TST37   024776  6606#  12462
TST4    007704  4339#  12435
```

```
TST40    025622          6730#  12463
TST41    026160          6806#  12464
TST42    026620          6900#  12465
TST43    027044          6958#  12466
TST44    027402          7056#  12467
TST45    027704          7129#  12468
TST46    030206          7202#  12469
TST47    030534          7282#  12470
TST5     010570          4473#  12436
TST50    031056          7362#  12471
TST51    031404          7443#  12472
TST52    031732          7523#  12473
TST53    032260          7602#  12474
TST54    032646          7698#  12475
TST55    033262          7797#  12476
TST56    033674          7896#  12477
TST57    034234          7975#  12478
TST6     011126          4550#  12437
TST60    034576          8057#  12479
TST61    035144          8141#  12480
TST62    035616          8242#  12481
TST63    036274          8351#  12482
TST64    037002          8464#  12483
TST65    037362          8560#  12484
TST66    037750          8647#  12485
TST67    040354          8741#  12486
TST7     011332          4597#  12438
TST70    040734          8821#  12487
TST71    041300          8903#  12488
TST72    041646          8987#  12489
TYPBN  = 104406          9264   12878#
TYPDS  = 104405          9104   9111   9258   12877#
TYPE   = 104401          3970   3990   4018   4040   4043   4049   4054   4057   4060   4064   4067   4075   4078
                         4084   4089   4094   4100   4110   4115   4121   4134   4135   4138   4144   4147   4154
                         9098   9105   9112   9147   9148   9156   9165   9171   9180   9218   9224   9228   9241
                         9247   9249   9267   9269   12053  12125  12191  12260  12320  12511  12519  12583  12595
                        12649  12650  12653  12664  12674  12685  12704  12752  12758  12763  12767  12772  12773
                        12775  12778  12782  12873#  12922
TYPOC  = 104402          4077   9261   12652  12874#
TYPON  = 104404         12876#
TYPOS  = 104403          4020   4091   4112   9151   9159   9168   9174   12875#
UBUSQS   066353          4064   12989#
UNS    = 040000          1720#  4422   4634   4637   4640   5010   5687   6756   7989   8039   8835   8885   10941
                        11064  11068  11072  11378  11410  11415
UNTMSK= 000007           1921#  6924   9773   9775
UPE    = 020000          1903#  4587   10157
USE    = 040000          1849#
USRFIL   105116         12989#
U0     = 000001          1918#
U1     = 000002          1917#
U2     = 000004          1916#
VV     = 000100          1714#  4648   4825   4828   9390   10083  10695  10760  10761  10810  10815  11036  11039
                        11252  11291  11292  11315  11321  11462  11711  11755  11756  11792  11797  11865  11866
                        11886  11891
WC     = 000040          1786#  4437   10951  11503
WCD    = 000050          1674#
```

```
WCE     = 040000            1902#   4587   10056   12989
WCEHI   = 010000            1927#
WCELO   = 004000            1928#
WCF     = 000040            1729#   1736   10127
WCH     = 000052            1675#
WD      = 000060            1678#
WH      = 000062            1679#
WLE     = 004000            1723#   1736    6580    6582   10108   10112   10125   10140   10155   12989
WRL     = 004000            1709#   4645    4756    4759    4807    4810    4811   10110
XDPMG     067143            3990   12989#
XSIZ      006004            4000#   4141
ZEROS     067466           12989#
$APTHD    001100            1987   1993#
$ASTAT=   ****** U         12965   12980
$ATYC     065754           12936   12938#
$ATY1     065730           12934#
$ATY3     065736           12305   12935#
$ATY4     065746           12524   12937#
$AUTOB    001150            2025#   3980*   4201*   12642   12799
$BASE     001276            2094#   4006    4076    4086*   4220    4260    4304    4347    4477    4554    4601    4667    4714
                            4861    4939    4997    5073    5145    5208    5275    5338    5401    5476    5575    5653    5736
                            5851    5925    6001    6087    6156    6271    6380    6495    6610    6734    6810    6904    6962
                            7064    7137    7210    7290    7370    7451    7531    7606    7702    7801    7900    7979    8065
                            8145    8246    8355    8468    8564    8651    8745    8825    8907    8995    9545    9618    9695
                           10572   10856   12989
$BDADR    001136            2020#   12989
$BDDAT    001142            2022#   4399*   4400*   4402    4407*   4411*   4416    4421*   4422*   4426*   4432*   4436*   4437*
                            4439    4444*   4448*   4449*   4451    4456*   4457*   4460*   4461*   4463    4580*   4582*   4586*
                            4587*   4622*   4628*   4636*   4642*   4649*   4655*   4741*   4746*   4747*   4752*   4753*   4758*
                            4759*   4764*   4765*   4770*   4771*   4792*   4797*   4798*   4803*   4804*   4809*   4810*   4815*
                            4816*   4821*   4822*   4827*   4828*   4848*   4850*   5435*   5436*   5441*   5442*   5504*   5505*
                            5540*   5541*   5602*   5603*   5632*   5633*   5680*   5681*   5711*   5712*   5960*   5961*   5965*
                            5966*   6041*   6042*   6048*   6053*   6121*   6122*   6143*   6144*   6242*   6243*   6244    6253*
                            6254*   6255    6351*   6352*   6353    6362*   6363*   6364    6466*   6467*   6468    6477*   6478*
                            6479    6581*   6582*   6583    6592*   6593*   6594    6705*   6706*   6707    6716*   6717*   6718
                            6894*   6948*   7007*   7008*   7009    7951*   7952*   8013*   8014*   8224*   8511*   8512*   8607*
                            8608*   8694*   8695*   8798*   8799*   8860*   8861*   9772*   9773*   9776    9794*   9800*   9816*
                            9833*   9849*   9877*   9927*   9928*   9940*   9941*   9951*   9952*   9954    9972*   9973*   9974
                            9994*   9995*   9996   10027*  10028*  10029   10043*  10044*  10045   10057*  10058*  10070*  10071*
                           10087*  10088*  10113*  10114*  10128*  10129*  10143*  10144*  10158*  10159*  10171*  10172*  10201*
                           10202*  10216*  10217*  10231*  10232*  10248*  10249*  10274*  10275*  10289*  10290*  10304*  10305*
                           10319*  10320*  10342*  10343*  10357*  10358*  10379*  10380*  10398*  10419*  10470*  10471*  10472
                           10497*  10498*  10499   10513*  10514   10525*  10526   10643*  10670*  10671*  10672   10692*  10703*
                           10704*  10710   10721   10734*  10747*  10772*  10786*  10787*  10800*  10816*  10900*  10901*  10903
                           10913*  10922*  10930   10940*  10941*  10950*  10951*  10953   10963*  10972*  10973*  10975   10985*
                           10986*  10995*  10996*  10998   11040*  11054*  11070*  11084*  11098*  11112*  11126*  11181*  11195*
                           11211*  11228*  11249*  11266*  11280*  11305*  11320*  11335*  11351*  11368*  11386*  11400*  11416*
                           11449*  11450*  11452   11461*  11462*  11464   11474*  11483*  11490*  11493   11502*  11503*  11505
                           11514*  11515*  11517   11527*  11536*  11613*  11631*  11662*  11663*  11664   11684*  11685*  11688
                           11708*  11725*  11741*  11764*  11782*  11798*  11812*  11876*  11892*  11908*  11928*  11942*  11998*
                           12013*  12989
$BELL     001212            2045#   12511   12544   12595   12792
$BIN      062244           12121*  12124*  12125   12132#
$CDW1     001302            2096#
$CDW2     001304            2097#
$CHARC    063176           12322*  12332*  12339   12348*  12353#
```

```
$CKSWR  064402          12628* 12882
$CMTAG  001114           2008*  3922   3923   3931   3937   3938   3939
$CM3   = 000000           2038*
$CM4   = 000005           2038*  2039*  2040*  2041*  2042*  2043*
$CNTLC  065300          12583  12664  12792*
$CNTLG  065312          12649  12794*
$CNTLU  065305          12674  12767  12793*
$CPUOP  001250           2068*
$CRLF   001217           2047*  9112   9147   9180   9218   9228   9241   9249   9269  12321  12356  12519  12544
                        12685  12772  12792
$DBLK   062462          12157  12191  12199*
$DDW0   001306           2098*
$DDW1   001310           2099*
$DDW2   001312           2100*
$DDW3   001314           2101*
$DDW4   001316           2102*
$DDW5   001320           2103*
$DDW6   001322           2104*
$DDW7   001324           2105*
$DEVCT  001232           2059*
$DEVM   001300           2095*  4003*  4015*  4133*  4142*  4160*  4161   4168
$DOAGN  042404           9094   9115   9121*
$DTBL   062452          12160  12195*
$ENDAD  042374           1966   3968   9117*  12539
$ENDCT  042240           3937   9096*
$ENULL  042410           9124*
$ENV    001242           2064*  3974   3995   4195  12300  12521  12943  12967
$ENVM   001243           2065*  3960   4000  12302  12307  12945
$EOP    042204           4249   9086*  12056
$EOPCT  042232           3937*  9093*  9097
$EOSP   042150           4292   4332   4651   8981   9068*
$ERFLG  001117           2011*  12361  12401  12403  12409*  12431  12503*  12544
$ERMAX  001131           2017*  3940*  12403  12426*  12431
$ERROR  063704           3931   12504*
$ERRPC  001132           2018*  9172   12513*  12514*  12515  12544
$ERRTB  001562           2206*  9187
$ERTTL  001126           2015*  9109   9113*  12512*  12544
$ESCAP  001210           2044*  3939*  12425*  12535  12537  12544
$ETABL  001242           2063*
$ETEND  001326           1999   2106*
$FATAL  001224           2056*  12971*
$FFLG   066174          12934*  12937*  12965  12974*  12982*
$FILLC  001172           2036*  12325  12356
$FILLS  001171           2035*  12356
$GDADR  001134           2019*  12989
$GDDAT  001140           2021*  4401*  4402   4406*  4415*  4416   4420*  4438*  4439   4443*  4450*  4451   4455*
                         4462*  4463   4581*  4619*  4620*  4621*  4629*  4637*  4643*  4647*  4648*  4656*  4740*
                         4748*  4754*  4760*  4766*  4772*  4791*  4799*  4805*  4811*  4817*  4823*  4829*  4849*
                         5437*  5443*  5503*  5542*  5604*  5634*  5682*  5710*  5959*  5967*  6044*  6049*  6054*
                         6123*  6145*  6240*  6241*  6244   6248*  6251*  6252*  6255   6349*  6350*  6353   6357*
                         6360*  6361*  6364   6464*  6465*  6468   6472*  6475*  6476*  6479   6579*  6580*  6583
                         6587*  6590*  6591*  6594   6703*  6704*  6707   6711*  6714*  6715*  6718   6893*  6947*
                         7006*  7009   7950*  8015*  8223*  8510*  8606*  8693*  8797*  8859*  9774*  9775*  9776
                         9792*  9793*  9799*  9801*  9813*  9814*  9815*  9831*  9832*  9847*  9848*  9875*  9876*
                         9929*  9942*  9953*  9954   9966*  9971*  9974   9985*  9993*  9996  10022*  10025*  10026*
                        10029  10041*  10042*  10045  10052*  10404*  10410*  10411*  10412*  10413*  10416*  10417  10464*
```

```
                    10469*  10472   10481*  10494*  10499   10510*  10511*  10512*  10514   10523*  10524*  10526   10642*
                    10654*  10667*  10672   10691*  10705*  10706*  10733*  10746*  10770*  10771*  10785*  10798*  10799*
                    10814*  10815*  10902*  10903   10912*  10928*  10930   10939*  10952*  10953   10962*  10974*  10975
                    10984*  10997*  10998   11038*  11039*  11052*  11053*  11071*  11072*  11085*  11086*  11099*  11100*
                    11113*  11114*  11127*  11128*  11179*  11180*  11193*  11194*  11209*  11210*  11226*  11227*  11247*
                    11248*  11264*  11265*  11278*  11279*  11303*  11304*  11319*  11321*  11333*  11334*  11349*  11350*
                    11366*  11367*  11384*  11385*  11398*  11399*  11414*  11415*  11451*  11452   11463*  11464   11473*
                    11504*  11505   11516*  11517   11525*  11611*  11612*  11629*  11630*  11645*  11658*  11664   11686*
                    11687*  11688   11692   11706*  11707*  11723*  11724*  11739*  11740*  11762*  11763*  11780*  11781*
                    11796*  11797*  11810*  11811*  11874*  11875*  11890*  11906*  11907*  11926*  11927*  11940*  11941*
                    11999*  12014*  12989
$GET42  042364       9114*
$GTSWR  064472      12650*  12880
$HD     = 000001     1505    1506
$HIBTS  001100       1994*
$HIOCT  065442      12832*  12837*
$ICNT   001120       2012*  12416*  12417   12419*  12430
$ILLUP  065712      12892   12908   12925*
$INTAG  001151       2026*  12647*  12666   12686   12799
$ITEMB  001130       2016*   9163   12515*  12523   12544
$LF     001220       2048*  12356   12544   12782   12792
$LFLG   066173      12975*  12981*
$LPADR  001122       2013*   3941*   4216*   4343*   4935*   4993*   5069*   5141*   5204*   7060*   7133*   7206*   7286*
                     7366*   7447*   7527*   8061*   8991*  12398*  12407*  12423*  12428   12430
$LPERR  001124       2014*   3942*   4217*   4344*   4936*   4994*   5070*   5142*   5205*   7061*   7134*   7207*   7287*
                     7367*   7448*   7528*   8062*   8992*  12407   12424*  12430   12534
$MADR1  001254       2081*
$MADR2  001260       2085*
$MADR3  001264       2088*
$MADR4  001270       2091*
$MAIL   001222       1995    1999    2054*   3959    3974    4195    4222    4262    4306    4349    4479    4556    4603
                     4669    4716    4863    4941    4999    5075    5147    5210    5277    5340    5403    5478    5577
                     5655    5738    5853    5927    6003    6089    6158    6273    6382    6497    6612    6736    6812
                     6906    6964    7066    7139    7212    7292    7372    7453    7533    7608    7704    7803    7902
                     7981    8067    8147    8248    8357    8470    8566    8653    8747    8827    8909    8997   12300
                    12422   12521
$MAMS1  001252       2075*
$MAMS2  001256       2083*
$MAMS3  001262       2086*
$MAMS4  001266       2089*
$MBADR  001102       1995*
$MFLG   066172      12935*  12941   12976*  12980*
$MNEW   065330      12653   12797*
$MSGAD  001236       2061*  12951*  12954
$MSGLG  001240       2062*  12956*
$MSGTY  001222       2055*  12949   12957*  12969   12973*
$MSWR   065317      12650   12795*
$MTYP1  001253       2076*
$MTYP2  001257       2084*
$MTYP3  001263       2087*
$MTYP4  001267       2090*
$MXCNT  063516      12420   12430*
$NULL   001170       2034*  12327   12356
$NWTST= 000001       4211*   4213    4252*   4254    4296*   4298    4335*   4337    4469*   4471    4546*   4548    4593*
                     4595    4660*   4707*   4853*   4855    4927*   4929    4986*   5062*   5134*   5197*   5267*   5269
                     5330*   5332    5393*   5395    5469*   5568*   5646*   5729*   5844*   5918*   5920    5993*   5995
```

```
                    6080#  6149#  6264#  6373#  6488#  6603#  6727#  6803#  6897#  6955#  7052#  7054   7125#
                    7127   7198#  7200   7278#  7280   7358#  7360   7439#  7441   7519#  7521   7599#  7695#
                    7794#  7893#  7972#  8054#  8138#  8239#  8348#  8461#  8557#  8644#  8738#  8818#  8900#
                    8984#
$OCNT   062714     12232#  12261#  12274#
$OMODE  062716     12227#  12231#  12236   12239#  12250#  12276#
$OVER   063502     12374   12399   12408   12418   12427#
$PASS   001230      2058#   3959#   9090#   9091#   9102    9124   12414   12431
$PASTM  001106      1997#
$POWER  065720     12923   12928#
$PWRDN  065552      3935   12892#  12920
$PWRMG  065706     12923#
$PWRUP  065624     12902   12908#
$QUES   001216      2046#  12356   12544   12704   12775   12792
$RDCHR  064744     12717#  12883
$RDDEC= ****** U   12886
$RDLIN  065034     12740#  12884
$RDOCT  065342     12810#  12885
$RDSZ = 000010     12733#
$RESRE  062134     12097#  12887
$RTNAD  042406      9123#
$R2A  = ****** U   12888
$SAVRE  062076     12081#  12886
$SAVR6  065716     12901#  12909   12910#  12911#  12927#
$SCOPE  063202      3929   12370#
$SETUP= 000137      1948#   3928    3929    3931    3933    3935    3937    3938    3939    3941    3968    3971    4192
                    9088   12371   12505   12531   12539   12587   12592   12593   12623   12799
$STUP = 177777      1948#
$SVLAD  063446     12382   12421#
$SVPC = 000200      1964#   1969
$SWR  = 167400      1494#   1505    1510    1511    1512    1513    1514    1515    1516    1517    2043    2044    2045
                    3938    3939    3941    3942    4222    4262    4306    4349    4479    4556    4603    4669    4716
                    4863    4941    4999    5075    5147    5210    5277    5340    5403    5478    5577    5655    5738
                    5853    5927    6003    6089    6158    6273    6382    6497    6612    6736    6812    6906    6964
                    7066    7139    7212    7292    7372    7453    7533    7608    7704    7803    7902    7981    8067
                    8147    8248    8357    8470    8566    8653    8747    8827    8909    8997    9083    9089    9116
                    9122    9124   12362   12363   12364   12365   12366   12373   12385   12387   12388   12401   12402
                   12403   12410   12411   12412   12424   12427   12430   12496   12497   12498   12499   12500   12509
                   12516   12528   12532   12544   12924
$SWREG  001244      2066#   3962
$SWRMK= 000000      1517    1518   12366   12367   12391
$SWO8T  063520     12397   12431#
$TESTN  001226      2057#   4222#   4262#   4306#   4349#   4479#   4556#   4603#   4669#   4716#   4863#   4941#   4999#
                    5075#   5147#   5210#   5277#   5340#   5403#   5478#   5577#   5655#   5738#   5853#   5927#   6003#
                    6089#   6158#   6273#   6382#   6497#   6612#   6736#   6812#   6906#   6964#   7066#   7139#   7212#
                    7292#   7372#   7453#   7533#   7608#   7704#   7803#   7902#   7981#   8067#   8147#   8248#   8357#
                    8470#   8566#   8653#   8747#   8827#   8909#   8997#   9155   12422#
$TIMES  001206      2043#   3938#   4342#   4934#   4992#   5068#   5140#   5203#   7059#   7132#   7205#   7285#   7365#
                    7446#   7526#   8060#   8990#   9089#  12410#  12417   12420#  12430
$TKB    001162      2031#  12547   12568   12579   12604   12632   12659
$TKCNT  064104     12548#  12563#  12593   12610#  12724   12726#
$TKINT  064114      4032    4204    4390   12563#  12584   12645
$TKQEN= 064113     12552#  12618   12729
$TKQIN  064106     12549#  12564#  12565   12616#  12617#  12618   12620#
$TKQOU  064110     12550#  12565#  12727   12728#  12729   12731#
$TKQSR  064112     12551#  12564   12620   12731
```

```
$TKS     001160          2030#  12547   12569*  12600*  12602   12608*  12630   12646*  12656   12668*  12688*
$TKSRV   064164         12566   12579#
$TMP0    001174          2038#   4276*   4279    6949*   7011*   9550*   9623*  10578*  10579   12989
$TMP1    001176          2039#   4042*   4043    4044    4047    4056*   4057    4058    4066*   4067    4070    4080*   4082
                         4086    4087*   4088*   4090    4096*   4098    4102    4103*   4104*   4105*   4106*   4107*   4108*
                         4109*   4111    4117*   4119    4124*   4125*   4126*   4127*   4128*   4129    4137*   4138    4139
                         4146*   4147    4148    4150    4152    4156    4275*   4283    4329*   9551*   9552    9624*   9625
                        10577*  10584   11983*  11984*  11996   11998   12989
$TMP2    001200          2040#   4330*  11986*  11987*  12010   12013   12989
$TMP3    001202          2041#
$TMP4    001204          2042#
$TN    = 000073          1505#   4211    4222*   4252    4262*   4296    4306*   4335    4349*   4469    4479*   4546    4556*
                         4593    4603*   4660    4669*   4707    4716*   4853    4863*   4927    4941*   4986    4999*   5062
                         5075*   5134    5147*   5197    5210*   5267    5277*   5330    5340*   5393    5403*   5469    5478*
                         5568    5577*   5646    5655*   5729    5738*   5844    5853*   5918    5927*   5993    6003*   6080
                         6089*   6149    6158*   6264    6273*   6373    6382*   6488    6497*   6603    6612*   6727    6736*
                         6803    6812*   6897    6906*   6955    6964*   7052    7066*   7125    7139*   7198    7212*   7278
                         7292*   7358    7372*   7439    7453*   7519    7533*   7599    7608*   7695    7704*   7794    7803*
                         7893    7902*   7972    7981*   8054    8067*   8138    8147*   8239    8248*   8348    8357*   8461
                         8470*   8557    8566*   8644    8653*   8738    8747*   8818    8827*   8900    8909*   8984    8997*
                        12392   12432
$TPB     001166          2033#  12345*  12356
$TPFLG   001173          2037#  12294   12356
$TPS     001164          2032#  12343   12356
$TRAP    065444          3933   12846#
$TRAP2   065504         12861#  12872
$TRP   = 000016         12865#  12874#  12875#  12876#  12877#  12878#  12879#  12880   12881#  12882   12883#  12884#  12885#
                        12886#  12887#  12888#
$TRPAD   065516         12855   12872#
$TSTM    001104          1996#
$TSTNM   001116          2010#   9088*  12361   12394*  12421*  12422   12427   12431   12508   12544
$TTYIN   065270         12742   12743   12755   12773   12787   12791#
$TYPBN   062172         12118#  12878
$TYPDS   062246         12145#  12877
$TYPE    062720         12294#  12865   12873   12962
$TYPEC   063132         12324   12331   12338   12343#  12344   12690
$TYPEX   063200         12349   12351   12354#
$TYPOC   062516         12230#  12874
$TYPON   062532         12229   12232#  12876
$TYPOS   062472         12225#  12875
$UNIT    001234          2060#   4209*   9149    9774
$UNITM   001110          1998#
$USWR    001246          2067#
$VECT1   001272          2092#   4087    4102*   4103    4129*   5747    5753    5861    5867
$VECT2   001274          2093#
$XTSTR   063220         12376#
$$GET4=  000000          9116#
$$SWD8=  000073         12431#  12432   12433#  12434#  12435#  12436#  12437#  12438#  12439#  12440#  12441#  12442#  12443#
                        12444#  12445#  12446#  12447#  12448#  12449#  12450#  12451#  12452#  12453#  12454#  12455#  12456#
                        12457#  12458#  12459#  12460#  12461#  12462#  12463#  12464#  12465#  12466#  12467#  12468#  12469#
                        12470#  12471#  12472#  12473#  12474#  12475#  12476#  12477#  12478#  12479#  12480#  12481#  12482#
                        12483#  12484#  12485#  12486#  12487#  12488#  12489#  12490#
$OFILL   062715         12226*  12230*  12240   12275#
$40CAT=  ****** U       12373   12518
.      = 111605          1951#   1955#   1964    1965#   1967#   1969#   1974#   1977#   1983    1984#   1986#   1988#   2000
                         2007#   2049    2167#   2183#   2188#   3926    3941    3942    9101#   9124    9125#  12063#  12199#
```