RM80 FCTNL TEST PT 1 CZRNDA0 AH-T111A-MC FICHE 1 OF 2 JUL 1982

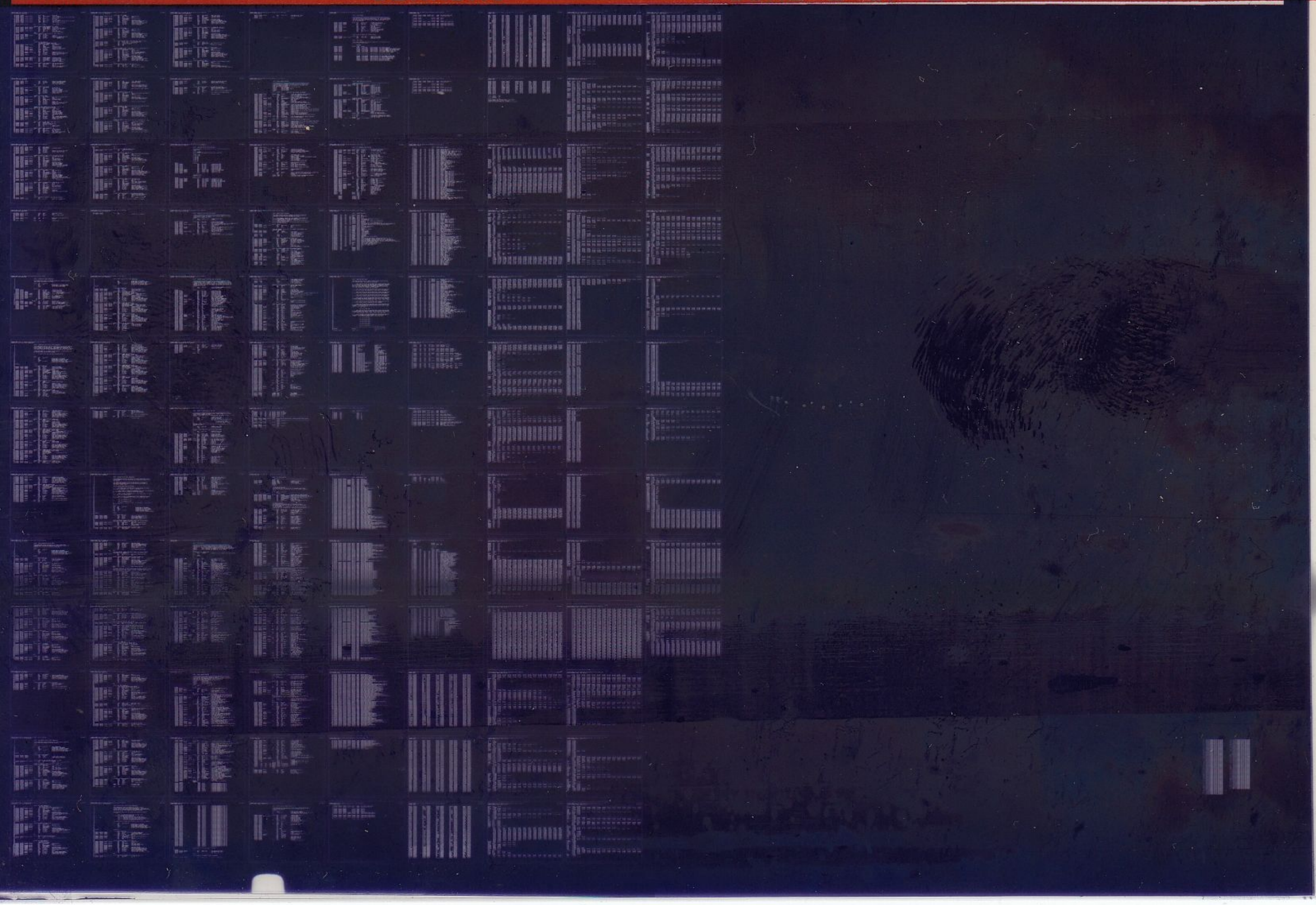
RM80

FCTNL TEST PT 1
CZRNDAO

AH-T111A-MC
FICHE 2 OF 2

JUL 1982

MADE IN USA



```
.REM    \
```

IDENTIFICATION

PRODUCT CODE: AC-T110A-MC

PRODUCT NAME: CZRNDA0 RM80 FCTNL TEST PT 1

PRODUCT DATE: APRIL 1, 1982

MAINTAINER: CX DIAGNOSTIC GROUP

AUTHOR: MIKE LEAVITT

CONTENTS

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM80 SUBSYSTEM FUNCTIONAL TEST IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE RM80 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 RM80 SUBSYSTEM, INCLUDING THE MASSBUS CONTROLLER, MASSBUS ADAPTER AND THE DISK DRIVE.

THE TEST IS COMPRISED OF 4 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; PART 4 TESTS MECHANICAL POSITIONING OPERATIONS AND VARIOUS TIMING PARAMETERS OF THE RM80 DISK DRIVE.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM80 DISK SUBSYSTEM WHICH CONSISTS OF THE RH70 MASSBUS CONTROLLER ,THE RM80 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 RM80 SUBSYSTEM FUNCTIONAL TEST:

PDP-11 PROCESSOR
20K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH70 CONTROLLER
1 TO 8 RM80 DISK DRIVES

2.2 MEDIA REQUIREMENTS

EACH UNIT BEING TESTED MUST BE LOADED WITH A DISK BEFORE TESTING BEGINS ON THAT UNIT. THE DISK MAY BE FORMATTED OR UNFORMATTED.

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM80 DISKLESS TEST, PART 1 & 2

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.

3.2 SWITCH OPTIONS

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

SW15 HALT ON ERROR
SW14 LOOP ON TEST (CURRENTLY BEING EXECUTED)
SW13 INHIBIT ERROR TYPEOUTS
SW12 UNUSED
SW11 INHIBIT TEST ITERATIONS
SW10 BELL ON ERROR
SW09 LOOP ON ERROR
SW08 LOOP ON TEST IN SW07-00

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

3.3 STARTING

THE PROGRAM MAY BE STARTED AT LOCATION 200 OR 204. STARTING AT 200 WILL BE THE NORMAL STARTING ADDRESS. STARTING AT 204 WILL ENABLE THE RH/RM BASE ADDRESS TO BE CHANGED. 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 OR BY PRESSING THE HALT SWITCH ON THE PROCESSOR FRONT PANEL.

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200 OR 204. (SEE SECTION 3.3)

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 (L) N ?". IF THE OPERATOR RESPONDS WITH A "Y", THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHICH WILL LIST SWITCH OPTIONS, ETC. ANY OTHER RESPONSE TO THE QUESTION IS CONSIDERED A 'N' AND NO HELP TEXT IS TYPED. THIS QUESTION IS ONLY ASKED ON THE INITIAL PROGRAM START AND NOT ON SUBSEQUENT START-UP'S.

ON THE PROGRAM INITIAL START AND WHEN RESTARTING AT LOCATION 204, THE OPERATOR MAY CHANGE THE RH/RM BASE ADDRESSES WITH THE FOLLOWING DIALOGUE.

EXAMPLE 1

```
RMCS1=176700 <CR>        ;NO CHANGE IN ADDRESS
RMVEC=000254 <CR>        ;NO CHANGE IN ADDRESS
```

EXAMPLE 2

```
RMCS1=176700 177200<CR> ;CHANGE BASE ADDRESS TO 177200
RMVEC=000254 260<CR>    ;CHANGE VECTOR ADDRESS TO 260
```

THE NEXT QUESTION ASKED IS 'DO YOU WANT MANUAL INTERVENTION TESTS (L) N ?'. IF THE OPERATOR ANSWERED WITH A 'Y' RESPONSE, UPON ENTERING A MANUAL INTERVENTION TEST, THE PROGRAM WILL ASK THE OPERATOR TO PERFORM CERTAIN FUNCTIONS TO THE DRIVE. THERE IS ONLY ONE MANUAL INTERVENTION TEST IN THIS PROGRAM. (SEE SECTION 6.0, TEST #2) ANY OTHER RESPONSE TO THE QUESTION IS CONSIDERED TO BE 'N' AND THERE WILL BE NO MANUAL INTERVENTION IN THE PROGRAM OPERATION. TERMINATE RESPONSE WITH A CARRIAGE RETURN.

ON THE INITIAL START, THE NEXT QUESTION TYPED IS, 'TYPE "A" TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S) AND TERMINATE INPUT WITH A CARRIAGE RETURN'. THEN, 'DRIVE(S):' IS TYPED AND WAITS FOR THE

OPERATOR TO TYPE AN "A", TO TEST ALL POSSIBLE DRIVES OR TYPE ANY STRING OF DRIVE NUMBER(S) TO BE TESTED AND TERMINATE THE INPUT WITH A "CARRIAGE RETURN". NO COMMAS OR ANY OTHER SEPARATORS ARE NEEDED WHEN ENTERING THE DRIVE NUMBERS AS A STRING. THE PROGRAM ENTERS THE COMMA SEPARTOR AUTOMATICALLY AFTER TYPING EACH NUMBER. ON ALL SUBSEQUENT STARTS, ONLY THE 'DRIVE(S):' PROMPT IS TYPED.

THE DIAGNOSTIC THEN INITIALIZES AND REPORTS THE STATUS OF THE DRIVES WHICH WHERE PREVIOUSLY SPECIFIED FOR TESTING. THE FOLLOWING IS AN EXAMPLE PRINTOUT:

```
'UNIT STATUS:
  0    ONLINE   RM80
  1    LOAD DEVICE
  2    OFFLINE  RM80
  3    NOT PRESENT
  4    NOT PRESENT
  5    NOT AN RM80
  6    NOT PRESENT
  7    NOT PRESENT'
```

THE ABOVE UNIT STATUS SHOWS THAT DRIVE 0 WILL BE TESTED, WHILE DRIVES 1 - 7 WILL NOT BE TESTED.

THE DIAGNOSTIC THEN TYPES THE FOLLOWING MESSAGE, BASED ON THE STATUS OF THE DRIVE:

```
'DRIVE(S) TO BE TESTED, 0'
```

IF NO DRIVES ARE AVAILABLE FOR TESTING THE FOLLOWING MESSAGE WILL BE TYPED TO THE OPERATOR:

```
'DRIVE(S) TO BE TESTED, NONE'
```

THE PROGRAM WILL THEN, EITHER START TESTING THE DRIVES AVAILABLE FOR TESTING OR RETURN TO THE BEGINNING OF THE PROGRAM AND WAIT.

ONCE THE DRIVES START TESTING, THE FOLLOWING MESSAGE WILL OCCUR AS EACH DRIVE BEGINS TO BE TESTED:

```
'DRIVE 0'
```

AFTER ALL THE DRIVES ARE COMPLETELY TESTED, THE END OF PASS MESSAGE WILL BE TYPED (SEE SECTION 4.3) AND THE PROGRAM WILL START TESTING ALL THE DRIVES AGAIN. THIS WILL CONTINUE UNTIL THE PROGRAM IS HALTED BY THE OPERATOR.

NOTE: THE LETTER LOCATED WITHIN THE BRACKETS () INDICATES THE TYPE OF RESPONSE REQUIRED BY THE USER, D=DECIMAL, O=OCTAL AND L=LETTER.

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 IS AS FOLLOWS.

```
229
230                         'END OF PASS      1'
231
232                     THE FOLLOWING MESSAGE WILL ALSO  OCCUR IF THERE WERE ERRORS SINCE
233                 THE LAST END OF PASS REPORT.
234
235                         'TOTAL ERRORS SINCE LAST REPORT      0'
236
237
238                 4.4  PERFORMANCE REPORT
239
240                     NO PERFORMANCE REPORTS ARE GIVEN  DURING  THE  EXECUTION  OF  THE
241                 PROGRAM.
242
243
244
245                 4.5  PROGRAM HALTS
246
247                     THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM.
248                 PROCESSOR HALTS ARE DUE TO THE TRAP CATCHER.
249
250
251
252                 4.6  ERROR REPORTS
253
254                     THE FIRST LINE OF THE ERROR REPORT CONTAINS  THE  NUMBER  OF  THE
255                 UNIT (DRIVE) BEING TESTED,  THE DRIVE TYPE, THE TEST NUMBER, THE ERROR
256                 NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED.
257                 THIS LINE IS FOLLOWED BY THE ERROR MESSAGE:  ONE OR MORE LINES OF TEXT
258                 WHICH GIVE A BRIEF, YET COMPREHENSIVE DESCRIPTION OF  THE  ERROR.  THE
259                 ERROR  MESSAGE  IS NORMALLY  FOLLOWED BY  ONE OR  MORE PAIRS  OF LINES
260                 CONTAINING DATA HEADERS AND DATA PERTININENT TO  THE ERROR,  INCLUDING
261                 EXPECTED AND ACTUAL TEST RESULTS.
262
263                 THE FOLLOWING PRINTOUT SHOWS A TYPICAL ERROR MESSAGE FOR THIS PROGRAM:
264
265                     DRV# 0 - RM80, TEST# 14, ERR# 326, PC=016654
266                     MASSBUS DATA BUS PARITY ERROR 'MDPE' (RMCS2, BIT 8) DETECTED
267                     DURING WRITE COMMAND
268                     EXPECTD    RECEVD
269                     040300     040700
270                     RMCS1      RMCS2   RMDS    RMER1   RMER2   RMAS
271                     144252     040700  010700  000000  000000  000000
272                     RMWC       RMBA    RMDA    RMOF    RMDC    RMEC1   RMEC2
273                     177403     104604  000002  010000  000000  004066  000000
274                     RMMR1      RMMR2   RMDT    RMSN
275                     000010     011777  024026  177777
276
277
278
279                 4.7  EXECUTION TIME
280
281                     PASS 1 OF THE PROGRAM TAKES ABOUT 20 SECONDS.  PASS 2 AND
282                 SUBSEQUENT PASSES TAKE 2.3 MINUTES.
283
284
285                 5.0  ENVIRONMENTAL SUPPORT
```

5.1 PROCESSOR COMPATIBILITY

THE RM80 SUBSYSTEM FUNCTIONAL TEST IS EXECUTABLE ON A PDP-11/70 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 RM80 SUBSYSTEM FUNCTIONAL TEST DOES NOT SPECIFICALLY TEST DUAL PORT LOGIC IN THE RM80 ADAPTER BUT IS EXECUTABLE ON RM80 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 RM80 SUSBYSTEM FUNCTIONAL TEST.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE IS NOT USED DURING THE RM80 SUSBYSTEM FUNCTIONAL TEST.

5.5 ACT11, APT11 COMPATIBILITY

THE RM80 SUBYSTEM 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 RM80 SUSBYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES, AND PROVIDES MEDIA PROTECTION IN THE CASE WHERE THE RM80 IS THE XXDP LOADING DEVICE.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

TEST 1 CONTROLLER ACCESS TEST

PURPOSE:

TO VERIFY THAT THE UNIBUS ADDRESS OF THE RM80 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.

TEST 2 LOGICAL ADDRESS PLUG TEST (OPTIONAL, SEE SECTION 4.2)

PURPOSE:

TO VERIFY THAT ALL LOGICAL ADDRESS PLUGS(0-7) FUNCTION IN EACH DEVICE UNDER TEST.

PROCEDURE:

THIS TEST SELECTS EACH UNIT BY INSERTING ADDRESS PLUGS 0-7 INTO EACH DEVICE BEING TESTED AND READS RMER2 TO VERIFY THAT "OPE" IS SET. A MASSBUS CLEAR IS ISSUED AND RMER2 IS READ TO VERIFY THAT "OPE" IS CLEAR. THE CONTROL STATUS REGISTERS 1 AND 2 ARE READ TO VERIFY THAT THE CORRECT UNIT WAS SELECTED AND IS AVAILABLE FOR TESTING, "DVA" IS SET IN RMCS1 AND "NED" IS CLEAR IN RMCS2.

TEST 3 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

RMER2 - 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.

TEST 4 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.

TEST 5 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.

TEST 6 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 RMER2 =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.

TEST 7 PRIMARY/SECONDARY ERROR TEST

PURPOSE:

TO VERIFY THAT THE RM80 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.

TEST 10 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.

TEST 11 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.

TEST 12 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.

TEST 13 ABORT RECALIBRATE TEST

PURPOSE:

TO VERIFY THAT THE RM80 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.

TEST 14 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.

TEST 15 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.

TEST 16 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.

TEST 17 DRIVE CLEAR TEST

PURPOSE:

TO VERIFY THAT ALL APPLICABLE RM80 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;

```
685                                     .DMD, DIAGNOSTIC MODE IS SET;
686
687                                     .RMER2, ERROR REGISTER 3, IS SET TO ALL ONES.
688
689                                FOLLOWING THE DRIVE CLEAR COMMAND, THE FOLLOWING  ITEMS  ARE
690                           CHECKED:
691
692                                     .RMCS1, IS CHECKED FOR DVA=1, F0-F4 AND GO=0, ALL OTHER BITS
693                           ARE DONT CARES;
694
695                                     .RMDS, IS CHECKED FOR 0,  EXCEPT  FOR  MOL,DPR,DRY,  AND  VV
696                           WHICH SHOULD BE ONE, AND PGM WHICH IS A DONT CARE.
697
698                                     .RMER1, IS CHECKED FOR 0;
699
700                                     .RMAS, IS CHECKED TO INSURE THE APPROPRIATE ATA BIT IS 0;
701
702                                     .RMMR, IS CHECKED FOR 0, EXCEPT FOR WORD CLOCK, LAST SECTOR,
703                           AND LAST SECTOR AND TRACK WHICH ARE DONT CARES;
704
705                                     .RMMR2 IS CHECKED FOR 0, EXCEPT FOR THE TEST  BIT  WHICH  IS
706                           ONE, AND REQA, REQB WHICH ARE DONT CARES;
707
708                                     .RMEC2 IS CHECKED FOR 0;
709
710                                     .RMER2 IS CHECKED FOR 0;
711
712
713
714
715
716                           TEST 20 NOP TEST
717
718                                PURPOSE:
719
720                                     TO VERIFY THE EXECUTION OF ''NOP'' COMMAND.
721
722                                PROCEDURE:
723
724                                     A NOP COMMAND IS EXECUTED ON THE SELECTED DEVICE AND  STATUS
725                           IS  CHECKED  TO  VERIFY  THAT  THERE WERE NO ERRORS OR UNEXPECTED
726                           CHANGES IN STATUS.
727
728
729
730
731
732                           TEST 21 OFFSET TEST
733
734                                PURPOSE:
735
736                                     TO VERIFY THE EXECUTION OF ''OFFSET'' COMMAND.
737
738                                PROCEDURE:
739
740                                     THE OFFSET COMMAND IS EXECUTED AND THE PROGRAM  CHECKS  THAT
741                           OFFSET STATUS, BIT 0 OF RMDS IS SET AND THAT ATTENTION, BIT 15 OF
```

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

TEST 22 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.

TEST 23 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.

TEST 24 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.

```
                        TEST 25 PROGRAM INTERRUPT TEST

                             PURPOSE:

                                  TO VERIFY THAT THE RM80 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.



                        TEST 26 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.



                    TEST 27 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.
```

```
                        TEST 30 READ IN PRESET TEST

                             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 017200) AND
                             VERIFIED TO BE ZERO AFTER THE TEST;



                        TEST 31 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.



                        TEST 32 ILLEGAL FUNCTION TEST

                             PURPOSE:

                                  TO VERIFY  THAT  THE  RM80  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.
```

```
913
914
915
916                         TEST 33 INVALID COMMAND TEST
917
918                              PURPOSE:
919
920                                   TO VERIFY IVC ERROR DETECTION.
921
922                              PROCEDURE:
923
924                                   THE TEST RESETS VOLUME VALID USING MAINTENANCE  UNIT  READY,
925                              THEN  EXECUTES  A  NOP COMMAND AND VERIFIES THAT IVC IS SET.  THE
926                              PROCESS IS REPEATED  FOR  EACH  FUNCTION  CODE,  WITH  IVC  BEING
927                              CHECKED ACCORDING TO THE FUNCTION.
928
929
930
931
932
933                         TEST 34 INVALID ADDRESS ERROR TEST
934
935                              PURPOSE:
936
937                                   TO VERIFY IAE ERROR DETECTION.
938
939                              PROCEDURE:
940
941                                   THE TEST EXECUTES EACH FUNCTION CODE WITH RMDA, AND RMDC SET
942                              TO ILLEGAL ADDRESSES, AND VERIFIES IAE ACCORDING TO THE FUNCTION.
943
944
945
946
947
948                         TEST 35 WRITE LOCK ERROR TEST
949
950                              PURPOSE:
951
952                                   TO VERIFY WLE ERROR DETECTION.
953
954                              PROCEDURE:
955
956                                   THE TEST SIMULATES WRITE  PROTECT  USING  MAINTENANCE  WRITE
957                              PROTECT   AND  VERIFIES  WLE  ACCORDING  TO  THE  FUNCTION  BEING
958                              EXECUTED.  EACH FUNCTION CODE IS TESTED.
959
960
961
962
963
964                         TEST 36 ERROR ABORT TESTS
965
966                              PURPOSE:
967
968                                   TO TEST COMMAND EXECUTION DURING AN ABORT CONDITION.
969
```

```
                        PROCEDURE:

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



                    TEST 37 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.



                    TEST 40 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 RM80 FOR EACH PATTERN.



                    TEST 41 ILLEGAL REGISTER TEST

                        PURPOSE:

                             TO VERIFY THE DETECTION OF ILLEGAL REGISTER ADDRESSES BY THE
                        RM80 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 (42 - 52)

PURPOSE:

THE PURPOSE OF EACH OF THE FOLLOWING SEEK TESTS IS TO VERIFY THE EXECUTION OF SEEK OPERATIONS BY THE RM80 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.

TEST 42 SEEK TO FIRST CYLINDER TEST

THIS TEST SEEKS TO THE FIRST CYLINDER.

TEST 43 SEEK TO LAST CYLINDER TEST

THIS TEST SEEKS TO THE LAST CYLINDER.

TEST 44 SEEK PRIME CYLINDERS TEST

THIS TEST SEEKS FORWARD TO EACH PRIME CYLINDER ADDRESS, I.E. CYLINDERS 1, 2, 4, 8, 16, 32, 64, 128, 256 AND 512.

TEST 45 SEEK ZERO DIFFERENCE TEST

THIS TEST EXECUTES SUCCESSIVE SEEKS TO CYLINDER 0.

TEST 46 SEEK MAXIMUM DIFFERENCE FORWARD

THIS TEST DOES A RECALIBRATE COMMAND, REGUARDLESS OF THE CONDITION OF "PIP" OR "SKI", AND THEN EXCUTES A SEEK TO THE LAST CYLINDER.

TEST 47 SEEK ADJACENT FORWARD TEST

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

TEST 50 SEEK ADJACENT REVERSE TEST

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

TEST 51 SEEK TO INVALID SECTOR TEST

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. THE TEST IS REPEATED FOR 16 BIT FORMAT.

TEST 52 SEEK TO INVALID TRACK TEST

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.

TEST 53 SEEK TO INVALID CYLINDER TEST

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.

TEST 54 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.

TEST 55 ABORT SEEK TEST

PURPOSE:

TO VERIFY THAT THE RM80 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.

TEST 56 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.

TEST 57 LOOK AHEAD TEST

PURPOSE:

1198
1199 TO INSURE THAT THE SECTOR COUNT WHICH ORIGINATES AT THE
1200 DRIVE AND IS VISIBLE THROUGH THE LOOK AHEAD REGISTER (RMLA) IS
1201 OPERATIONAL.
1202
1203 PROCEDURE:
1204
1205 WITH THE OFFSET REGISTER SET FOR 18 BIT FORMAT, THE PROGRAM
1206 SAMPLES THE LOOK AHEAD REGISTER AND COLLECTS EACH DIFFERENT
1207 SAMPLE UNTIL THE VALUE OF THE FIRST SAMPLE IS DETECTED OR UNTIL
1208 31. SAMPLES ARE TAKEN. THE COLLECTION IS THEN TESTED TO DETERMINE
1209 THAT THE SECTOR COUNT INCREMENTS CORRECTLY THROUGH THE ENTIRE
1210 RANGE OF VALID SECTORS. THE SAME PROCEDURE IS REPEATED FOR 16
1211 BIT FORMAT, WHERE THE LIMIT ON THE NUMBER OF SAMPLES IS 32. ALSO,
1212 16 BIT FORMAT WITH SSEI SET, WHERE THE LIMIT ON THE NUMBER OF
1213 SAMPLES IS 33.
1214
1215
1216
1217
1218
1219 TEST 60 SEARCH ON CYLINDER
1220
1221 PURPOSE:
1222
1223
1224 TO VERIFY THE EXECUTION OF SEARCH OPERATIONS WITH NO HEAD
1225 MOTION USING THE SECTOR PULSE FOR SECTOR COMPARE.
1226
1227 PROCEDURE:
1228
1229 THE TEST INTIALIZES AND RECALIBRATES THE DRIVE IF "PIP" OR
1230 "SKI" IS ACTIVE THEN SEEKS TO CYLINDER 0. THE TEST THEN DOES A
1231 SEARCH TO EACH SECTOR AND VERIFIES THAT THE SEARCH COMPLETES
1232 WITHOUT ERROR. THIS TEST IS DONE IN 16 BIT FORMAT WITH SSEI SET.
1233
1234
1235
1236
1237
1238 TEST 61 SEARCH OFF CYLINDER
1239
1240 PURPOSE:
1241
1242 TO VERIFY THE EXECUTION OF A SEARCH COMMAND WITH IMPLIED
1243 HEAD MOTION.
1244
1245 PROCEDURE:
1246
1247 THE DRIVE IS RECALIBRATED AND INITIALIZED IF THE DRIVE IS
1248 OFF CYLINDER OR HAS A SEEK ERROR. THE PROGRAM THEN EXECUTES AN
1249 EXPLICIT SEEK TO CYLINDER 279., FOLLOWED BY A SEARCH TO CYLINDER
1250 280., TRACK 0, SECTOR 0. THE NEXT SECTOR IS SEARCHED FOR BY AN
1251 EXPLICIT SEEK TO CYLINDER 278., FOLLOWED BY A SEARCH TO CYLINDER
1252 281., TRACK 0, SECTOR 1., ETC., UNTIL ALL SECTORS HAVE BEEN
1253 SEARCHED. THE TEST IS DONE FOR 16 BIT FORMAT WITH SSEI SET, WHERE
1254 THE LAST SECTOR SEARCHED IS 31.

TEST 62 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. THIS TEST IS EXECUTED IN 16 BIT FORMAT, WHERE SECTOR 31. IS THE INVALID SECTOR

TEST 63 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.

TEST 64 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.

TEST 65 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.

TEST 66 ABORT SEARCH TEST

PURPOSE:

TO VERIFY THAT THE RM80 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.

TEST 67 SEARCH AT OFFSET TEST

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.

```
                        ;*LAST REVISION 19-OCT-81
                        .TITLE  CZRNDAO RM80 FCTNL PT1
                        ;*COPYRIGHT (C) 1982
                        ;*DIGITAL EQUIPMENT CORPORATION
                        ;*COLORADO SPGS., CO. 80919
                        ;*
                        ;*PROGRAM BY MIKE LEAVITT
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH              USE
                        ;*      ------              ---
                        ;*        15                HALT ON ERROR
                        ;*        14                LOOP ON TEST
                        ;*        13                INHIBIT ERROR TYPEOUTS
                        ;*        12                UNUSED
                        ;*        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
104000                  ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE   = IOT           ;;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
177776                  PSW=PS
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
000003                  R3      = %3            ;;GENERAL REGISTER
```

BASIC DEFINITIONS

```
000004                  R4      = %4            ;;GENERAL REGISTER
000005                  R5      = %5            ;;GENERAL REGISTER
000006                  R6      = %6            ;;GENERAL REGISTER
000007                  R7      = %7            ;;GENERAL REGISTER
000006                  SP      = %6            ;;STACK POINTER
000007                  PC      = %7            ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0     = 0             ;;PRIORITY LEVEL 0
000040                  PR1     = 40            ;;PRIORITY LEVEL 1
000100                  PR2     = 100           ;;PRIORITY LEVEL 2
000140                  PR3     = 140           ;;PRIORITY LEVEL 3
000200                  PR4     = 200           ;;PRIORITY LEVEL 4
000240                  PR5     = 240           ;;PRIORITY LEVEL 5
000300                  PR6     = 300           ;;PRIORITY LEVEL 6
000340                  PR7     = 340           ;;PRIORITY LEVEL 7

                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15    = 100000
040000                  SW14    = 40000
020000                  SW13    = 20000
010000                  SW12    = 10000
004000                  SW11    = 4000
002000                  SW10    = 2000
001000                  SW09    = 1000
000400                  SW08    = 400
000200                  SW07    = 200
000100                  SW06    = 100
000040                  SW05    = 40
000020                  SW04    = 20
000010                  SW03    = 10
000004                  SW02    = 4
000002                  SW01    = 2
000001                  SW00    = 1
001000                  SW9=SW09
000400                  SW8=SW08
000200                  SW7=SW07
000100                  SW6=SW06
000040                  SW5=SW05
000020                  SW4=SW04
000010                  SW3=SW03
000004                  SW2=SW02
000002                  SW1=SW01
000001                  SW0=SW00

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15   = 100000
040000                  BIT14   = 40000
020000                  BIT13   = 20000
010000                  BIT12   = 10000
004000                  BIT11   = 4000
002000                  BIT10   = 2000
001000                  BIT09   = 1000
000400                  BIT08   = 400
000200                  BIT07   = 200
000100                  BIT06   = 100
000040                  BIT05   = 40
```

```
                000020          BIT04   = 20
                000010          BIT03   = 10
                000004          BIT02   = 4
                000002          BIT01   = 2
                000001          BIT00   = 1
                001000          BIT9=BIT09
                000400          BIT8=BIT08
                000200          BIT7=BIT07
                000100          BIT6=BIT06
                000040          BIT5=BIT05
                000020          BIT4=BIT04
                000010          BIT3=BIT03
                000004          BIT2=BIT02
                000002          BIT1=BIT01
                000001          BIT0=BIT00

                                ;*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
    480
    481                         .SBTTL  RM80 REGISTER BIT DEFINITIONS
    482
    483                         ;*RMCS1 CONTROL STATUS REGISTER
    484
    485         004000          DVA     = BIT11                 ;DEVICE AVAILABLE-READ ONLY
    486         000040          F4      = BIT05                 ;FUNCTION CODE
    487         000020          F3      = BIT04                 ;FUNCTION CODE
    488         000010          F2      = BIT03                 ;FUNCTION CODE
    489         000004          F1      = BIT02                 ;FUNCTION CODE
    490         000002          F0      = BIT01                 ;FUNCTION CODE
    491         000001          GO      = BIT00                 ;GO BIT
    492         000077          FNCMSK  = 000077                ;FUNCTION CODE MASK
    493
    494                         ;FUNCTION CODES (BITS 01-05 OF RMCS1)
    495         000000          NOP     = 000000                ;NOP COMMAND
    496         000002          ILF02   = 000002                ;ILLEGAL COMMAND
    497         000004          SEEK    = 000004                ;SEEK COMMAND
    498         000006          RECAL   = 000006                ;RECALIBRATE COMMAND
    499         000010          DRVCLR  = 000010                ;DRIVE CLEAR COMMAND
    500         000012          RLEASE  = 000012                ;RELEASE COMMAND
    501         000014          OFFSET  = 000014                ;OFFSET COMMAND
    502         000016          RTC     = 000016                ;RETURN TO CENTERLINE COMMAND
    503         000020          RIP     = 000020                ;READ IN PRESET COMMAND
    504         000022          PAKACK  = 000022                ;PACK ACKNOWLEDGE COMMAND
    505         000022          PACACK  = PAKACK
    506         000024          ILF24   = 000024                ;ILLEGAL COMMAND
    507         000026          ILF26   = 000026                ;ILLEGAL COMMAND
```

RM80 REGISTER BIT DEFINITIONS

```
                        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

                        ;TRACK ADDRESS DEFINITIONS
004000                  TA8     = BIT11                 ;TRACK ADDRESS 8.
002000                  TA4     = BIT10                 ;TRACK ADDRESS 4
001000                  TA2     = BIT09                 ;TRACK ADDRESS 2
000400                  TA1     = BIT08                 ;TRACK ADDRESS 1

                        ;SECTOR ADDRESS DEFINITIONS
000020                  SA16    = BIT04                 ;SECTOR ADDRESS 16.
000010                  SA8     = BIT03                 ;SECTOR ADDRESS 8.
000004                  SA4     = BIT02                 ;SECTOR ADDRESS 4
000002                  SA2     = BIT01                 ;SECTOR ADDRESS 2
000001                  SA1     = BIT00                 ;SECTOR ADDRESS 1

                        ;TRACK & SECTOR MASKS
177400                  TADMSK  = 177400                ;TRACK ADDRESS MASK
000377                  SADMSK  = 000377                ;SECTOR ADDRESS MASK

                        ;*RMDS  DRIVE STATUS REGISTER

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

                        ;*RMER1 ERROR REGISTER #1
```

```
RM80 REGISTER BIT DEFINITIONS

                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

                                ;*RMMR1 MAINTENANCE REGISTER #1

                                ;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

                                ;READ ONLY BITS
                100000          OCC     = BIT15                 ;OCCUPIED
                040000          RG      = BIT14                 ;RUN AND GO
```

RM80 REGISTER BIT DEFINITIONS

```
617          020000                  EBL     = BIT13                 ;END OF BLOCK
618          010000                  REX     = BIT12                 ;EXCEPTION
619          004000                  ESRC    = BIT11                 ;ENABLE SEARCH
620          002000                  PLFS    = BIT10                 ;LOOKING FOR SYNC
621          001000                  ECRC    = BIT09                 ;ENABLE CRC OUT
622          000400                  PDA     = BIT08                 ;DATA AREA
623          000200                  PHA     = BIT07                 ;HEADER AREA
624          000100                  CONT    = BIT06                 ;CONTINUE
625          000040                  WC      = BIT05                 ;WORD CLOCK
626          000020                  EECC    = BIT04                 ;ENABLE ECC OUT
627          000010                  MWD     = BIT03                 ;WRITE DATA BIT
628          000004                  LS      = BIT02                 ;LAST SECTOR
629          000002                  LST     = BIT01                 ;LAST SECTOR AND TRACK
630          000001                  DMD     = BIT00                 ;DIAGNOSTIC MODE
631
632                                  ;*RMDT  DRIVE TYPE REGISTER
633
634          100000                  NSA     = BIT15                 ;NOT SECTOR ADDRESSED = 0
635          040000                  TAP     = BIT14                 ;TAPE DRIVE = 0
636          020000                  MOH     = BIT13                 ;MOVING HEAD = 1
637          004000                  DRQ     = BIT11                 ;DRIVE REQUEST REQUIRED
638
639                                  ;*RMOF  OFFSET REGISTER
640
641          010000                  FMT16   = BIT12                 ;16 BIT WORD FORMAT
642          004000                  ECI     = BIT11                 ;ECC INHIBIT
643          002000                  HCI     = BIT10                 ;HEADER COMPARE INHIBIT
644          001000                  SSEI    = BIT09                 ;SKIP SECTOR ERROR INHIBIT
645          000200                  OFD     = BIT07                 ;OFFSET FORWARD
646
647                                  ;*RMDC  DESIRED CYLINDER ADDRESS REGISTER
648
649          001777                  CYLMSK  = 001777                ;MASK FOR CYLINDER ADDRESS
650
651                                  ;*RMMR2 MAINTENANCE REGISTER #2
652
653                                  ;READ ONLY BITS
654          100000                  RQA     = BIT15                 ;PORT A REQUEST
655          040000                  RQB     = BIT14                 ;PORT B REQUEST
656          020000                  TAG     = BIT13                 ;TAG CONTROL
657          010000                  TST     = BIT12                 ;COMMAND SEQUENCE TEST BIT
658          004000                  CC      = BIT11                 ;CONTROL OR CYLINDER TAG
659          002000                  CH      = BIT10                 ;CONTROL OR HEAD TAG
662          001000                  BB09    = BIT09                 ;TAG BUS
             000400                  BB08    = BIT08                 ;TAG BUS
             000200                  BB07    = BIT07                 ;TAG BUS
             000100                  BB06    = BIT06                 ;TAG BUS
             000040                  BB05    = BIT05                 ;TAG BUS
             000020                  BB04    = BIT04                 ;TAG BUS
             000010                  BB03    = BIT03                 ;TAG BUS
             000004                  BB02    = BIT02                 ;TAG BUS
             000002                  BB01    = BIT01                 ;TAG BUS
             000001                  BB00    = BIT00                 ;TAG BUS
663
664                                  ;*RMER2 ERROR REGISTER 2
665
666          100000                  BSE     = BIT15                 ;BAD SECTOR ERROR
```

RM80 REGISTER BIT DEFINITIONS

```
                        SKI     = BIT14                 ;SEEK INCOMPLETE
                        OPE     = BIT13                 ;OPERATOR PLUG ERROR
                        IVC     = BIT12                 ;INVALID COMMAND ERROR
                        LSC     = BIT11                 ;LOSS OF SYSTEM CLOCK
                        LBC     = BIT10                 ;LOSS OF BIT CLOCK
                        DVC     = BIT07                 ;DEVICE CHECK
                        SSE     = BIT05                 ;SKIP SECTOR ERROR
                        DPE     = BIT03                 ;DATA PARITY ERROR

                        .SBTTL  PROGRAM MNEMONICS

                        MSE     = BIT15                 ;MANUFACTURING DETECTED SECTOR ERROR
                        USE     = BIT14                 ;USER DETECTED SECTOR ERROR
                        SSF     = BIT13                 ;SKIP SECTOR FAILURE

                        .SBTTL  RM80 REGISTER INDEX VALUES

                        RMCS1   = 00                    ;CONTROL STATUS REGISTER #1
                        RMDA    = 06                    ;DISK ADDRESS REGISTER
                        RMDS    = 12                    ;DRIVE STATUS REGISTER
                        RMER1   = 14                    ;ERROR REGISTER #1
                        RMAS    = 16                    ;ATTENTION SUMMARY REGISTER
                        RMLA    = 20                    ;LOOK AHEAD REGISTER
                        RMMR1   = 24                    ;MAINTENANCE REGISTER
                        RMDT    = 26                    ;DRIVE TYPE REGISTER
                        RMSN    = 30                    ;SERIAL NUMBER REGISTER
                        RMOF    = 32                    ;OFFSET REGISTER
                        RMDC    = 34                    ;DESIRED CYLINDER REGISTER
                        RMHR    = 36                    ;HOLDING REGISTER
                        RMMR2   = 40                    ;MAINTENANCE REGISTER #2
                        RMER2   = 42                    ;ERROR REGISTER #2
                        RMEC1   = 44                    ;ECC POSITION REGISTER
                        RMEC2   = 46                    ;ECC PATTERN REGISTER
                        ILRG50  = 50                    ;ILLEGAL REGISTER 50
                        ILRG52  = 52                    ;ILLEGAL REGISTER 52
                        ILRG54  = 54                    ;ILLEGAL REGISTER 54
                        ILRG56  = 56                    ;ILLEGAL REGISTER 56
                        ILRG60  = 60                    ;ILLEGAL REGISTER 60
                        ILRG62  = 62                    ;ILLEGAL REGISTER 62
                        ILRG64  = 64                    ;ILLEGAL REGISTER 64
                        ILRG66  = 66                    ;ILLEGAL REGISTER 66
                        ILRG70  = 70                    ;ILLEGAL REGISTER 70
                        ILRG72  = 72                    ;ILLEGAL REGISTER 72
                        ILRG74  = 74                    ;ILLEGAL REGISTER 74
                        ILRG76  = 76                    ;ILLEGAL REGISTER 76

                        IDXMSK  = 77                    ;MASK FOR REGISTER INDEX NUMBER

                        .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS

                        ;*RMCS1 CONTROL STATUS REGISTER #1

                        SC      = BIT15                 ;SPECIAL CONDITION-READ ONLY
                        TRE     = BIT14                 ;TRANSFER ERROR
                        MCPE    = BIT13                 ;MASSBUS CONTROL BUS PARITY ERROR-READ ONLY
                        PSEL    = BIT10                 ;PORT B SELECT
                        A17     = BIT09                 ;ADDRESS EXTENSION
```

RH CONTROLLER REGISTER BIT DEFINITIONS

```
715          000400                  A16     = BIT08                 ;ADDRESS EXTENSION
716          000200                  RDY     = BIT07                 ;READY-READ ONLY
717          000100                  IE      = BIT06                 ;INTERRUPT ENABLE
718
719                                  ;*RMCS2 RH CONTROL STATUS REGISTER #2
720
721          100000                  DLT     = BIT15                 ;DATA LATE-READ ONLY
722          040000                  WCE     = BIT14                 ;WRITE CHECK ERROR-READ ONLY
723          020000                  UPE     = BIT13                 ;UNIBUS PARITY ERROR
724          010000                  NED     = BIT12                 ;NONEXISTANT DRIVE-READ ONLY
725          004000                  NEM     = BIT11                 ;NONEXISTANT MEMORY-READ ONLY
726          002000                  PGE     = BIT10                 ;PROGRAM ERROR-READ ONLY
727          001000                  MXF     = BIT09                 ;MISSED TRANSFER
728          000400                  MDPE    = BIT08                 ;MASSBUS DATA BUS PARITY ERROR-READ ONLY
729          000200                  OR      = BIT07                 ;OUTPUT READY-READ ONLY
730          000100                  IR      = BIT06                 ;INPUT READY-READ ONLY
731          000040                  CLR     = BIT05                 ;CONTROLLER CLEAR
732          000020                  PAT     = BIT04                 ;PARITY TEST
733          000010                  BAI     = BIT03                 ;UNIBUS ADDRESS INCREMENT INHIBIT
736          000004                  U2      = BIT02                 ;UNIT SELECT
             000002                  U1      = BIT01                 ;UNIT SELECT
             000001                  U0      = BIT00                 ;UNIT SELECT
737
738                                  ;UNIT SELECT MASK
739
740          000007                  UNTMSK  = 7                     ;UNIT SELECT MASK
741
742                                  ;*RMCS3 RH70 CONTROL STATUS REGISTER #3
743
744          100000                  APE     = BIT15                 ;ADDRESS PARITY ERROR
745          040000                  DPEHI   = BIT14                 ;DATA PARITY ERROR HIGH WORD
746          020000                  DPELO   = BIT13                 ;DATA PARITY ERROR LOW WORD
747          010000                  WCEHI   = BIT12                 ;WRITE CHECK ERROR HIGH WORD
748          004000                  WCELO   = BIT11                 ;WRITE CHECK ERROR LOW WORD
749          002000                  DBL     = BIT10                 ;DOUBLE WORD TRANSFER
750          000100                  IE      = BIT06                 ;INTERRUPT ENABLE
751          000010                  IPCK3   = BIT03                 ;INVERT PARITY CHECK
752          000004                  IPCK2   = BIT02                 ;INVERT PARITY CHECK
753          000002                  IPCK1   = BIT01                 ;INVERT PARITY CHECK
754          000001                  IPCK0   = BIT00                 ;INVERT PARITY CHECK
755
756                                  .SBTTL  RH CONTROLLER REGISTER INDEX VALUES
757
758          000000                  RMCS1   = 00                    ;CONTROL, STATUS REGISTER #1
759          000002                  RMWC    = 02                    ;WORD COUNT REGISTER
760          000004                  RMBA    = 04                    ;BUS ADDRESS REGISTER
761          000010                  RMCS2   = 10                    ;CONTROL, STATUS REGISTER #2
762          000022                  RMDB    = 22                    ;DATA BUFFER
763          000050                  RMBAE   = 50                    ;BUS ADDRESS EXTENSION
764          000052                  RMCS3   = 52                    ;CONTROL, STATUS REGISTER #3
765
766          176700                  ABASE   = 176700                ;UNIBUS ADDRESS
767          120254                  AVECT1  = 120254                ;UNIBUS VECTOR ADDRESS AND PRIORITY
768
```

```
                                .SBTTL  TRAP CATCHER

                000000                  .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174          .=174
  000174        000000  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
  000176        000000  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER

                                .SBTTL  STARTING ADDRESS(ES)

  000200  000137  005426        JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
2
3 000204  000137  005416        JMP     @#START1        ;CHANGE RH/RM BUS ADDRESS
4
5                               .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                000210          $SVPC=.                 ;SAVE PC
                000046          .=46
  000046        041512          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052          .=52
  000052        000000          .WORD   0               ;;2)SET LOC.52 TO ZERO
                000210          .=$SVPC                 ;; RESTORE PC
6
7               001100  .=1100
8                               .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
                001100          .$X=.   ;;SAVE CURRENT LOCATION
                000024          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
  000024        000200          200     ;;FOR APT START UP
                000044          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
  000044        001100          $APTHDR ;;POINT TO APT HEADER BLOCK
                001100          .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

  001100                        $APTHD:
  001100  000000                $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
  001102  001222                $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
  001104  000024                $TSTM:  .WORD   20.     ;;RUN TIM OF LONGEST TEST
  001106  000024                $PASTM: .WORD   20.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  001110  000024                $UNITM: .WORD   20.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
  001112  000042                        .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
9               001114  TAGADR=.
```

```
     0                                      .SBTTL  COMMON TAGS

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

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

                                            ;;*******************************************************************
                                            .EVEN
        001222                              $MAIL:                  ;;APT MAILBOX
        001222  000000                      $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
        001224  000000                      $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
        001226  000000                      $TESTN: .WORD   ATESTN  ;;TEST NUMBER
```

```
001230  000000                  $PASS:   .WORD   APASS   ;;PASS COUNT
001232  000000                  $DEVCT:  .WORD   ADEVCT  ;;DEVICE COUNT
001234  000000                  $UNIT:   .WORD   AUNIT   ;;I/O UNIT NUMBER
001236  000000                  $MSGAD:  .WORD   AMSGAD  ;;MESSAGE ADDRESS
001240  000000                  $MSGLG:  .WORD   AMSGLG  ;;MESSAGE LENGTH
001242                          $ETABLE:                 ;;APT ENVIRONMENT TABLE
001242     000                  $ENV:    .BYTE   AENV    ;;ENVIRONMENT BYTE
001243     000                  $ENVM:   .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001244  000000                  $SWREG:  .WORD   ASWREG  ;;APT SWITCH REGISTER
001246  000000                  $USWR:   .WORD   AUSWR   ;;USER SWITCHES
001250  000000                  $CPUOP:  .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                       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
001252     000                  $MAMS1:  .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001253     000                  $MTYP1:  .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                       MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001254  000000                  $MADR1:  .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                         MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001256     000                  $MAMS2:  .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001257     000                  $MTYP2:  .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001260  000000                  $MADR2:  .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001262     000                  $MAMS3:  .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001263     000                  $MTYP3:  .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001264  000000                  $MADR3:  .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001266     000                  $MAMS4:  .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001267     000                  $MTYP4:  .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001270  000000                  $MADR4:  .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001272  120254                  $VECT1:  .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001274  000000                  $VECT2:  .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001276  176700                  $BASE:   .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001300  000000                  $DEVM:   .WORD   ADEVM   ;;DEVICE MAP
001302  000000                  $CDW1:   .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001304  000000                  $CDW2:   .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001306  000000                  $DDW0:   .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001310  000000                  $DDW1:   .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001312  000000                  $DDW2:   .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001314  000000                  $DDW3:   .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001316  000000                  $DDW4:   .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001320  000000                  $DDW5:   .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001322  000000                  $DDW6:   .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001324  000000                  $DDW7:   .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001326                          $ETEND:
                                .MEXIT
```

```
                                .SBTTL  USER DEFINED TAGS

001326  000000          CHGADR: .WORD   0       ;CHANGE RH/RM BUS ADDRESS = -1, NO CHANGE = 0
001330  000000          MANUAL: .WORD   0       ;ENABLE MANUAL INTERVENTION TEST(S) = 1
001332  000000          XXDP:   .WORD   0       ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                                ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
                                                ;'XXDP' DEVICE CODE FOR THE RM80.
001334      000         LSTRK:  .BYTE   0       ;LO BYTE = 0
001335      000                 .BYTE   0       ;HI BYTE, CONTAINS LAST TRACK ADDRESS OF UNIT
                                                ;UNDER TEST. RM80 = 13.

                                ;THE REGISTER INPUT BUFFER IS USED FOR
                                ;STORING DRIVE STATUS

001336                          GETBUF:

                                ;REGISTER INPUT BUFFER
001336  000000          RMCS1I: .WORD   0       ;CONTROL, STATUS REGISTER #1
001340  000000          RMWCI:  .WORD   0       ;WORD COUNT REGISTER
001342  000000          RMBAI:  .WORD   0       ;BUS ADDRESS REGISTER
001344  000000          RMDAI:  .WORD   0       ;DISK ADDRESS REGISTER
001346  000000          RMCS2I: .WORD   0       ;CONTROL, STATUS REGISTER #2
001350  000000          RMDSI:  .WORD   0       ;DRIVE STATUS REGISTER
001352  000000          RMER1I: .WORD   0       ;ERROR REGISTER #1
001354  000000          RMASI:  .WORD   0       ;ATTENTION SUMMARY REGISTER
001356  000000          RMLAI:  .WORD   0       ;LOOK AHEAD REGISTER
001360  000000          RMDBI:  .WORD   0       ;DATA BUFFER
001362  000000          RMMR1I: .WORD   0       ;MAINTENANCE REGISTER #1
001364  000000          RMDTI:  .WORD   0       ;DRIVE TYPE REGISTER
001366  000000          RMSNI:  .WORD   0       ;SERIAL NUMBER REGISTER
001370  000000          RMOFI:  .WORD   0       ;OFFSET REGISTER
001372  000000          RMDCI:  .WORD   0       ;DESIRED CYLINDER REGISTER
001374  000000          RMHRI:  .WORD   0       ;HOLDING REGISTER
001376  000000          RMMR2I: .WORD   0       ;MAINTENANCE REGISTER #2
001400  000000          RMER2I: .WORD   0       ;ERROR REGISTER #2
001402  000000          RMEC1I: .WORD   0       ;ECC POSITION REGISTER
001404  000000          RMEC2I: .WORD   0       ;ECC PATTERN REGISTER
001406  000000          RMBAEI: .WORD   0       ;BUS ADDRESS EXTENSION REGISTER
001410  000000          RMCS3I: .WORD   0       ;CONTROL, STATUS REGISTER #3

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

001412                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001412  000000          RMCS1O: .WORD   0       ;CONTROL, STATUS REGISTER #1
001414  000000          RMWCO:  .WORD   0       ;WORD COUNT REGISTER
001416  000000          RMBAO:  .WORD   0       ;BUS ADDRESS REGISTER
001420  000000          RMDAO:  .WORD   0       ;DISK ADDRESS REGISTER
001422  000000          RMCS2O: .WORD   0       ;CONTROL, STATUS REGISTER #2
001424  000000          RMDSO:  .WORD   0       ;DRIVE STATUS REGISTER
001426  000000          RMER1O: .WORD   0       ;ERROR REGISTER #1
001430  000000          RMASO:  .WORD   0       ;ATTENTION SUMMARY REGISTER
001432  000000          RMLAO:  .WORD   0       ;LOOK AHEAD REGISTER
001434  000000          RMDBO:  .WORD   0       ;DATA BUFFER
001436  000000          RMMR1O: .WORD   0       ;MAINTENANCE REGISTER #1
```

```
001440  000000                  RMDTO:  .WORD   0       ;DRIVE TYPE REGISTER
001442  000000                  RMSNO:  .WORD   0       ;SERIAL NUMBER REGISTER
001444  000000                  RMOFO:  .WORD   0       ;OFFSET REGISTER
001446  000000                  RMDCO:  .WORD   0       ;DESIRED CYLINDER REGISTER
001450  000000                  RMHRO:  .WORD   0       ;HOLDING REGISTER
001452  000000                  RMMR2O: .WORD   0       ;MAINTENANCE REGISTER #2
001454  000000                  RMER2O: .WORD   0       ;ERROR REGISTER #2
001456  000000                  RMEC1O: .WORD   0       ;ECC POSITION REGISTER
001460  000000                  RMEC2O: .WORD   0       ;ECC PATTERN REGISTER
001462  000000                  RMBAEO: .WORD   0       ;BUS ADDRESS EXTENSION REGISTER
001464  000000                  RMCS3O: .WORD   0       ;CONTROL, STATUS REGISTER #3

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

001466  000000                  TSTQUE: .WORD   0               ;CONTAINS DEVICE POINTER
001470                                  .BLKW   8.              ;TEST QUE FOR DEVICES UNDER TEST
001510  000000                          .WORD   0               ;TABLE TERMINATOR GOES HERE WHEN
                                                                ;ALL 8. DEVICES ARE UNDER TEST.

001512  000000                  CLKADR: .WORD   0               ;UNIBUS ADDRESS OF KW11 CLOCK
001514  000000                  CLKVCT: .WORD   0               ;VECTOR ADDRESS OF KW11 CLOCK

                                ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
                                ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
                                ;A NEGATIVE BYTE.
001516                          GETINX: .BLKB   23.             ;GET INDEX TABLE

                                ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
                                ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
                                ;A NEGATIVE BYTE.
001545                          PUTINX: .BLKB   23.             ;PUT INDEX TABLE

                                ;PUT TAGS HERE
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


  001574                        $ERRTB:
1
2
                                ;ERROR  1       WRONG UNIT SELECTED
3
  001574  070112                        EMT1
  001576  074256                        EHT1
  001600  074406                        EDT1
  001602  074476                        EFT1

4
5
                                ;ERROR  2       DEVICE WENT UNAVAILABLE
6
  001604  070116                        EMT2
  001606  074256                        EHT1
  001610  074406                        EDT1
  001612  074476                        EFT1

7
8
                                ;ERROR  3       DEVICE WENT NONEXISTENT
9
  001614  070124                        EMT3
  001616  074256                        EHT1
  001620  074406                        EDT1
  001622  074476                        EFT1

10
11
                                ;ERROR  4       CONTROLLER NOT READY
12
  001624  070132                        EMT4
  001626  074256                        EHT1
  001630  074406                        EDT1
  001632  074476                        EFT1

13
14
                                ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
15
  001634  070140                        EMT5
  001636  074256                        EHT1
  001640  074406                        EDT1
  001642  074476                        EFT1

16
17
                                ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
```

ERROR POINTER TABLE

```
18
001644  070146                          EMT6
001646  074256                          EHT1
001650  074406                          EDT1
001652  074476                          EFT1

19
20                              ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
21
001654  070154                          EMT7
001656  000000                          0
001660  000000                          0
001662  000000                          0

22
23                              ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
24
001664  070162                          EMT10
001666  074256                          EHT1
001670  074406                          EDT1
001672  074476                          EFT1

25
26                              ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
27
001674  070166                          EMT11
001676  074256                          EHT1
001700  074406                          EDT1
001702  074476                          EFT1

28
29                              ;ERROR  12      INCORRECT FUNCTION CODE
30
001704  070172                          EMT12
001706  074256                          EHT1
001710  074406                          EDT1
001712  074476                          EFT1

31
32                              ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
33
001714  070200                          EMT13
001716  074256                          EHT1
001720  074406                          EDT1
001722  074476                          EFT1

34
35                              ;ERROR  14      TRANSFER ERROR IS INCORRECT
36
001724  070212                          EMT14
001726  074256                          EHT1
001730  074406                          EDT1
001732  074476                          EFT1

37
38                              ;ERROR  15      INCORRECT WORD COUNT
39
```

ERROR POINTER TABLE

```
001734  070220                          EMT15
001736  074256                          EHT1
001740  074406                          EDT1
001742  074476                          EFT1

40
41                              ;ERROR  16      INCORRECT BUS ADDRESS
42
001744  070226                          EMT16
001746  074256                          EHT1
001750  074406                          EDT1
001752  074476                          EFT1

43
44                              ;ERROR  17      INCORRECT LBT STATUS
45
001754  070236                          EMT17
001756  074256                          EHT1
001760  074406                          EDT1
001762  074476                          EFT1

46
47                              ;ERROR  20      INCORRECT AOE
48
001764  070246                          EMT20
001766  074256                          EHT1
001770  074406                          EDT1
001772  074476                          EFT1

49
50                              ;ERROR  21      INCORRECT DISK ADDRESS
51
001774  070256                          EMT21
001776  074256                          EHT1
002000  074406                          EDT1
002002  074476                          EFT1

52
53                              ;ERROR  22      INCORRECT CYLINDER ADDRESS
54
002004  070266                          EMT22
002006  074256                          EHT1
002010  074406                          EDT1
002012  074476                          EFT1

55
56                              ;ERROR  23      INCORRECT WLE STATUS
57
002014  070276                          EMT23
002016  074256                          EHT1
002020  074406                          EDT1
002022  074476                          EFT1

58
59                              ;ERROR  24      INCORRECT UPE STATUS
60
002024  070306                          EMT24
```

ERROR POINTER TABLE

```
        002026  074256                          EHT1
        002030  074406                          EDT1
        002032  074476                          EFT1

    61
    62                              ;ERROR  25      INCORRECT WCF STATUS
    63
        002034  070316                          EMT25
        002036  074256                          EHT1
        002040  074406                          EDT1
        002042  074476                          EFT1

    64
    65                              ;ERROR  26      INCORRECT WCE STATUS
    66
        002044  070326                          EMT26
        002046  074256                          EHT1
        002050  074406                          EDT1
        002052  074476                          EFT1

    67
    68                              ;ERROR  27      INCORRECT MDPE STATUS
    69
        002054  070336                          EMT27
        002056  074256                          EHT1
        002060  074406                          EDT1
        002062  074476                          EFT1

    70
    71                              ;ERROR  30      INCORRECT DCK STATUS
    72
        002064  070346                          EMT30
        002066  074256                          EHT1
        002070  074406                          EDT1
        002072  074476                          EFT1

    73
    74                              ;ERROR  31      INCORRECT ECH STATUS
    75
        002074  070356                          EMT31
        002076  074256                          EHT1
        002100  074406                          EDT1
        002102  074476                          EFT1

    76
    77                              ;ERROR  32      DLT SHOULD NOT BE SET
    78
        002104  070366                          EMT32
        002106  074256                          EHT1
        002110  074406                          EDT1
        002112  074476                          EFT1

    79
    80                              ;ERROR  33      MXF SHOULD NOT BE SET
    81
        002114  070376                          EMT33
        002116  074256                          EHT1
```

ERROR POINTER TABLE

```
002120  074406                          EDT1
002122  074476                          EFT1

                                ;ERROR  34      DTE SHOULD NOT BE SET

002124  070406                          EMT34
002126  074256                          EHT1
002130  074406                          EDT1
002132  074476                          EFT1

                                ;ERROR  35      INCORRECT HCRC STATUS

002134  070416                          EMT35
002136  074256                          EHT1
002140  074406                          EDT1
002142  074476                          EFT1

                                ;ERROR  36      INCORRECT HCE STATUS

002144  070426                          EMT36
002146  074256                          EHT1
002150  074406                          EDT1
002152  074476                          EFT1

                                ;ERROR  37      INCORRECT FER STATUS

002154  070436                          EMT37
002156  074256                          EHT1
002160  074406                          EDT1
002162  074476                          EFT1

                                ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)

002164  070446                          EMT40
002166  074256                          EHT1
002170  074406                          EDT1
002172  074476                          EFT1

                                ;ERROR  41      LOST 'MOL" DURING PACK ACKNOWLEDGE

002174  070454                          EMT41
002176  074256                          EHT1
002200  074406                          EDT1
002202  074476                          EFT1

                                ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE

002204  070464                          EMT42
002206  074256                          EHT1
002210  074406                          EDT1
```

ERROR POINTER TABLE

```
        002212  074476                          EFT1

103
104                                 ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
105
        002214  070476                          EMT43
        002216  074256                          EHT1
        002220  074406                          EDT1
        002222  074476                          EFT1

106
107                                 ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
108
        002224  070506                          EMT44
        002226  074256                          EHT1
        002230  074406                          EDT1
        002232  074476                          EFT1

109
110                                 ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
111
        002234  070516                          EMT45
        002236  074256                          EHT1
        002240  074406                          EDT1
        002242  074476                          EFT1

112
113                                 ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
114
        002244  070526                          EMT46
        002246  074256                          EHT1
        002250  074406                          EDT1
        002252  074476                          EFT1

115
116                                 ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
117
        002254  070536                          EMT47
        002256  074256                          EHT1
        002260  074406                          EDT1
        002262  074476                          EFT1

118
119                                 ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
120
        002264  070544                          EMT50
        002266  074256                          EHT1
        002270  074406                          EDT1
        002272  074476                          EFT1

121
122                                 ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
123
        002274  070554                          EMT51
        002276  074256                          EHT1
        002300  074406                          EDT1
        002302  074476                          EFT1
```

ERROR POINTER TABLE

```
                                        ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE

002304  070566                                  EMT52
002306  074256                                  EHT1
002310  074406                                  EDT1
002312  074476                                  EFT1

                                        ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
                                        ;               ON CYLINDER LATCH DIDN'T RESET

002314  070604                                  EMT53
002316  074256                                  EHT1
002320  074406                                  EDT1
002322  074476                                  EFT1

                                        ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND

002324  070622                                  EMT54
002326  074256                                  EHT1
002330  074406                                  EDT1
002332  074476                                  EFT1

                                        ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND

002334  070632                                  EMT55
002336  074256                                  EHT1
002340  074406                                  EDT1
002342  074476                                  EFT1

                                        ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET

002344  070644                                  EMT56
002346  074256                                  EHT1
002350  074406                                  EDT1
002352  074476                                  EFT1

                                        ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND

002354  070662                                  EMT57
002356  074256                                  EHT1
002360  074406                                  EDT1
002362  074476                                  EFT1

                                        ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
                                        ;               VOLUME VALID

002364  070672                                  EMT60
002366  074256                                  EHT1
002370  074406                                  EDT1
```

ERROR POINTER TABLE

```
        002372  074476                          EFT1

147
148                                     ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
149                                     ;               VOLUME VALID IS STIL SET
150
        002374  070710                          EMT61
        002376  074256                          EHT1
        002400  074406                          EDT1
        002402  074476                          EFT1

151
152                                     ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
153                                     ;               REPORTED DURING SEEK COMMAND
154
        002404  070730                          EMT62
        002406  074256                          EHT1
        002410  074406                          EDT1
        002412  074476                          EFT1

155
156                                     ;ERROR  63      SSE SHOULD NOT BE SET
157
        002414  070744                          EMT63
        002416  074256                          EHT1
        002420  074406                          EDT1
        002422  074476                          EFT1

158
159                                     ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
160
        002424  070754                          EMT64
        002426  074256                          EHT1
        002430  074406                          EDT1
        002432  074476                          EFT1

161
162                                     ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
163
        002434  070774                          EMT65
        002436  074256                          EHT1
        002440  074406                          EDT1
        002442  074476                          EFT1

164
165                                     ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
166
        002444  071014                          EMT66
        002446  074256                          EHT1
        002450  074406                          EDT1
        002452  074476                          EFT1

167
168                                     ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
169
        002454  071026                          EMT67
        002456  074256                          EHT1
```

ERROR POINTER TABLE

```
                002460  074406                          EDT1
                002462  074476                          EFT1

    170
    171                                 ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
    172
                002464  071040                          EMT70
                002466  074256                          EHT1
                002470  074406                          EDT1
                002472  074476                          EFT1

    173
    174                                 ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
    175
                002474  071050                          EMT71
                002476  074256                          EHT1
                002500  074406                          EDT1
                002502  074476                          EFT1

    176
    177                                 ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
    178
                002504  071060                          EMT72
                002506  074256                          EHT1
                002510  074406                          EDT1
                002512  074476                          EFT1

    179
    180                                 ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
    181                                 ;               CYLINDER DIDNT DROP
    182
                002514  071076                          EMT73
                002516  074256                          EHT1
                002520  074406                          EDT1
                002522  074476                          EFT1

    183
    184                                 ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
    185
                002524  071114                          EMT74
                002526  074256                          EHT1
                002530  074406                          EDT1
                002532  074476                          EFT1

    186
    187                                 ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
    188
                002534  071124                          EMT75
                002536  074256                          EHT1
                002540  074406                          EDT1
                002542  074476                          EFT1

    189
    190                                 ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
    191
                002544  071144                          EMT76
                002546  074256                          EHT1
```

ERROR POINTER TABLE

```
                002550  074406                          EDT1
                002552  074476                          EFT1

    192
    193                                 ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
    194
                002554  071154                          EMT77
                002556  074256                          EHT1
                002560  074406                          EDT1
                002562  074476                          EFT1

    195
    196                                 ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
    197
                002564  071166                          EMT100
                002566  074256                          EHT1
                002570  074406                          EDT1
                002572  074476                          EFT1

    198
    199                                 ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
    200
                002574  071204                          EMT101
                002576  074256                          EHT1
                002600  074406                          EDT1
                002602  074476                          EFT1

    201
    202                                 ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
    203
                002604  071222                          EMT102
                002606  074256                          EHT1
                002610  074406                          EDT1
                002612  074476                          EFT1

    204
    205                                 ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
    206
                002614  071232                          EMT103
                002616  074256                          EHT1
                002620  074406                          EDT1
                002622  074476                          EFT1

    207
    208                                 ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
    209
                002624  071244                          EMT104
                002626  074256                          EHT1
                002630  074406                          EDT1
                002632  074476                          EFT1

    210
    211                                 ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
    212
                002634  071262                          EMT105
                002636  074256                          EHT1
                002640  074406                          EDT1
```

ERROR POINTER TABLE

```
                002642  074476                          EFT1

    213
    214                                 ;ERROR  106     UNEXPECTED ''RMR'' ERROR DURING RECALIBRATE
    215
                002644  071272                          EMT106
                002646  074256                          EHT1
                002650  074406                          EDT1
                002652  074476                          EFT1

    216
    217                                 ;ERROR  107     ''UNS'' ERROR DURING RECALIBRATE - AC POWER IS LOW
    218
                002654  071302                          EMT107
                002656  074256                          EHT1
                002660  074406                          EDT1
                002662  074476                          EFT1

    219
    220                                 ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
    221
                002664  071322                          EMT110
                002666  074300                          EHT110
                002670  074424                          EDT110
                002672  074514                          EFT110

    222
    223                                 ;ERROR  111     NONEXISTENT DEVICE
    224
                002674  071334                          EMT111
                002676  074304                          EHT111
                002700  074426                          EDT111
                002702  074516                          EFT111

    225
    226                                 ;ERROR  112     DEVICE NOT AVAILABLE
    227
                002704  071342                          EMT112
                002706  074304                          EHT111
                002710  074426                          EDT111
                002712  074516                          EFT111

    228
    229                                 ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
    230
                002714  071350                          EMT113
                002716  000000                          0
                002720  000000                          0
                002722  000000                          0

    231
    232                                 ;ERROR  114     UNUSED
    233
                002724  000000                          0
                002726  000000                          0
                002730  000000                          0
                002732  000000                          0
```

ERROR POINTER TABLE

```
234
235                                     ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
236
     002734  071364                             EMT115
     002736  074256                             EHT1
     002740  074406                             EDT1
     002742  074476                             EFT1

237
238                                     ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
239
     002744  071374                             EMT116
     002746  074256                             EHT1
     002750  074406                             EDT1
     002752  074476                             EFT1

240
241                                     ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
242
     002754  071404                             EMT117
     002756  074256                             EHT1
     002760  074406                             EDT1
     002762  074476                             EFT1

243
244                                     ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
245
     002764  071414                             EMT120
     002766  074256                             EHT1
     002770  074406                             EDT1
     002772  074476                             EFT1

246
247                                     ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
248
     002774  071424                             EMT121
     002776  074256                             EHT1
     003000  074406                             EDT1
     003002  074476                             EFT1

249
250                                     ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
251
     003004  071434                             EMT122
     003006  074256                             EHT1
     003010  074406                             EDT1
     003012  074476                             EFT1

252
253                                     ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
254
     003014  071444                             EMT123
     003016  074256                             EHT1
     003020  074406                             EDT1
     003022  074476                             EFT1
```

ERROR POINTER TABLE

```
255
256                                     ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
257
    003024  071454                              EMT124
    003026  074256                              EHT1
    003030  074406                              EDT1
    003032  074476                              EFT1

258
259                                     ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
260
    003034  071464                              EMT125
    003036  074256                              EHT1
    003040  074406                              EDT1
    003042  074476                              EFT1

261
262                                     ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
263
    003044  071474                              EMT126
    003046  074256                              EHT1
    003050  074406                              EDT1
    003052  074476                              EFT1

264
265                                     ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
266
    003054  071506                              EMT127
    003056  074256                              EHT1
    003060  074406                              EDT1
    003062  074476                              EFT1

267
268                                     ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
269
    003064  071520                              EMT130
    003066  074256                              EHT1
    003070  074406                              EDT1
    003072  074476                              EFT1

270
271                                     ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
272
    003074  071532                              EMT131
    003076  074256                              EHT1
    003100  074406                              EDT1
    003102  074476                              EFT1

273
274                                     ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
275
    003104  071544                              EMT132
    003106  074256                              EHT1
    003110  074406                              EDT1
    003112  074476                              EFT1

276
```

ERROR POINTER TABLE

```
277                                     ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
278
     003114  071556                             EMT133
     003116  074256                             EHT1
     003120  074406                             EDT1
     003122  074476                             EFT1

279
280                                     ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
281
     003124  071570                             EMT134
     003126  074256                             EHT1
     003130  074406                             EDT1
     003132  074476                             EFT1

282
283                                     ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
284
     003134  071602                             EMT135
     003136  074256                             EHT1
     003140  074406                             EDT1
     003142  074476                             EFT1

285
286                                     ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
287
     003144  071614                             EMT136
     003146  074256                             EHT1
     003150  074406                             EDT1
     003152  074476                             EFT1

288
289                                     ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
290
     003154  071626                             EMT137
     003156  074256                             EHT1
     003160  074406                             EDT1
     003162  074476                             EFT1

291
292                                     ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
293
     003164  071636                             EMT140
     003166  074256                             EHT1
     003170  074406                             EDT1
     003172  074476                             EFT1

294
295                                     ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
296
     003174  071646                             EMT141
     003176  074256                             EHT1
     003200  074406                             EDT1
     003202  074476                             EFT1

297
298                                     ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
```

ERROR POINTER TABLE

```
299
     003204  071656                          EMT142
     003206  074256                          EHT1
     003210  074406                          EDT1
     003212  074476                          EFT1

300
301                                 ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
302
     003214  071666                          EMT143
     003216  074256                          EHT1
     003220  074406                          EDT1
     003222  074476                          EFT1

303
304                                 ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
305
     003224  071676                          EMT144
     003226  074256                          EHT1
     003230  074406                          EDT1
     003232  074476                          EFT1

306
307                                 ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
308
     003234  071706                          EMT145
     003236  074256                          EHT1
     003240  074406                          EDT1
     003242  074476                          EFT1

309
310                                 ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
311
     003244  071716                          EMT146
     003246  074256                          EHT1
     003250  074406                          EDT1
     003252  074476                          EFT1

312
313                                 ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
314
     003254  071726                          EMT147
     003256  074256                          EHT1
     003260  074406                          EDT1
     003262  074476                          EFT1

315
316                                 ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
317
     003264  071736                          EMT150
     003266  074256                          EHT1
     003270  074406                          EDT1
     003272  074476                          EFT1

318
319                                 ;ERROR  151     MEDIUM NOT ON LINE
320
```

ERROR POINTER TABLE

```
003274  071746                          EMT151
003276  074256                          EHT1
003300  074406                          EDT1
003302  074476                          EFT1

321
322                             ;ERROR  152     DRIVE FAULT
323
003304  071760                          EMT152
003306  074256                          EHT1
003310  074406                          EDT1
003312  074476                          EFT1

324
325                             ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
326
003314  071772                          EMT153
003316  074256                          EHT1
003320  074406                          EDT1
003322  074476                          EFT1

327
328                             ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
329
003324  072010                          EMT154
003326  074256                          EHT1
003330  074406                          EDT1
003332  074476                          EFT1

330
331                             ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
332
003334  072026                          EMT155
003336  074256                          EHT1
003340  074406                          EDT1
003342  074476                          EFT1

333
334                             ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
335
003344  072040                          EMT156
003346  074256                          EHT1
003350  074406                          EDT1
003352  074476                          EFT1

336
337                             ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
338
003354  072052                          EMT157
003356  074256                          EHT1
003360  074406                          EDT1
003362  074476                          EFT1

339
340                             ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
341
003364  072064                          EMT160
```

ERROR POINTER TABLE

```
003366  074256                          EHT1
003370  074406                          EDT1
003372  074476                          EFT1

342
343                             ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
344
003374  072074                          EMT161
003376  074256                          EHT1
003400  074406                          EDT1
003402  074476                          EFT1

345
346                             ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
347
003404  072106                          EMT162
003406  074256                          EHT1
003410  074406                          EDT1
003412  074476                          EFT1

348
349                             ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
350                             ;               WRITE BUFFER
351
003414  074052                          EMT336
003416  074344                          EHT336
003420  074442                          EDT336
003422  074532                          EFT336

352
353                             ;ERROR  164     OPI SHOULD NOT BE SET
354
003424  072130                          EMT164
003426  074256                          EHT1
003430  074406                          EDT1
003432  074476                          EFT1

355
356                             ;ERROR  165     IVC SHOULD NOT BE SET
357
003434  072136                          EMT165
003436  074256                          EHT1
003440  074406                          EDT1
003442  074476                          EFT1

358
359                             ;ERROR  166     IAE SHOULD NOT BE SET
360
003444  072144                          EMT166
003446  074256                          EHT1
003450  074406                          EDT1
003452  074476                          EFT1

361
362                             ;ERROR  167     NEM SHOULD NOT BE SET
363
003454  072152                          EMT167
```

ERROR POINTER TABLE

```
                003456  074256                          EHT1
                003460  074406                          EDT1
                003462  074476                          EFT1

    364
    365                                 ;ERROR  170     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
    366
                003464  072160                          EMT170
                003466  074256                          EHT1
                003470  074406                          EDT1
                003472  074476                          EFT1

    367
    368                                 ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
    369
                003474  072172                          EMT171
                003476  074256                          EHT1
                003500  074406                          EDT1
                003502  074476                          EFT1

    370
    371                                 ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
    372
                003504  072202                          EMT172
                003506  074256                          EHT1
                003510  074406                          EDT1
                003512  074476                          EFT1

    373
    374                                 ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
    375
                003514  072212                          EMT173
                003516  074256                          EHT1
                003520  074406                          EDT1
                003522  074476                          EFT1

    376
    377                                 ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
    378
                003524  072222                          EMT174
                003526  074256                          EHT1
                003530  074406                          EDT1
                003532  074476                          EFT1

    379
    380                                 ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
    381
                003534  072234                          EMT175
                003536  074256                          EHT1
                003540  074406                          EDT1
                003542  074476                          EFT1

    382
    383                                 ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
    384
                003544  072242                          EMT176
                003546  074256                          EHT1
```

ERROR POINTER TABLE

```
003550  074406                          EDT1
003552  074476                          EFT1

385
386                             ;ERROR  177     --RESERVED FOR POWER MONITOR BIT FAILURE--
387
003554  000000                          0
003556  000000                          0
003560  000000                          0
003562  000000                          0

388
389                             ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
390
003564  072252                          EMT200
003566  074256                          EHT1
003570  074406                          EDT1
003572  074476                          EFT1

391
392                             ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
393
003574  072264                          EMT201
003576  074256                          EHT1
003600  074406                          EDT1
003602  074476                          EFT1

394
395                             ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
396
003604  072276                          EMT202
003606  074256                          EHT1
003610  074406                          EDT1
003612  074476                          EFT1

397
398                             ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
399
003614  072310                          EMT203
003616  074256                          EHT1
003620  074406                          EDT1
003622  074476                          EFT1

400
401                             ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
402
003624  072322                          EMT204
003626  074256                          EHT1
003630  074406                          EDT1
003632  074476                          EFT1

403
404                             ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
405
003634  072340                          EMT205
003636  074256                          EHT1
003640  074406                          EDT1
```

ERROR POINTER TABLE

```
003642  074476                          EFT1

406
407                             ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
408
003644  072350                          EMT206
003646  074256                          EHT1
003650  074406                          EDT1
003652  074476                          EFT1

409
410                             ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
411
003654  072360                          EMT207
003656  074256                          EHT1
003660  074406                          EDT1
003662  074476                          EFT1

412
413                             ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
414
003664  072372                          EMT210
003666  074256                          EHT1
003670  074406                          EDT1
003672  074476                          EFT1

415
416                             ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
417
003674  072400                          EMT211
003676  074256                          EHT1
003700  074406                          EDT1
003702  074476                          EFT1

418
419                             ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
420
003704  072414                          EMT212
003706  074256                          EHT1
003710  074406                          EDT1
003712  074476                          EFT1

421
422                             ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
423
003714  072430                          EMT213
003716  074256                          EHT1
003720  074406                          EDT1
003722  074476                          EFT1

424
425                             ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
426
003724  072440                          EMT214
003726  074270                          EHT2
003730  074416                          EDT2
003732  074506                          EFT2
```

ERROR POINTER TABLE

```
427
428                                     ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
429
     003734  072460                               EMT215
     003736  074270                               EHT2
     003740  074416                               EDT2
     003742  074506                               EFT2

430
431                                     ;ERROR  216     INCORRECT "IVC" STATUS
432
     003744  072472                               EMT216
     003746  074256                               EHT1
     003750  074406                               EDT1
     003752  074476                               EFT1

433
434                                     ;ERROR  217     INCORRECT "IAE" STATUS
435
     003754  072502                               EMT217
     003756  074256                               EHT1
     003760  074406                               EDT1
     003762  074476                               EFT1

436
437                                     ;ERROR 220      INCORRECT "WLE" STATUS
438
     003764  072512                               EMT220
     003766  074256                               EHT1
     003770  074406                               EDT1
     003772  074476                               EFT1

439
440                                     ;ERROR  221     INCORRECT "OPI" STATUS
441
     003774  072522                               EMT221
     003776  074256                               EHT1
     004000  074406                               EDT1
     004002  074476                               EFT1

442
443                                     ;ERROR  222     RM80 DID NOT DETECT RMR ERROR
444
     004004  072532                               EMT222
     004006  074256                               EHT1
     004010  074406                               EDT1
     004012  074476                               EFT1

445
446                                     ;ERROR  223     RM80 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
447
     004014  072542                               EMT223
     004016  074320                               EHT223
     004020  074432                               EDT223
     004022  074522                               EFT223
```

ERROR POINTER TABLE

```
448
449                                     ;ERROR  224     SSE NOT DETECTED DURING DATA TRANSFER
450
004024  072552                                  EMT224
004026  074256                                  EHT1
004030  074406                                  EDT1
004032  074476                                  EFT1

451
452                                     ;ERROR  225     "SSEI" NOT RESET BY END OF TRACK DURING DATA TRANSFER
453
004034  072566                                  EMT225
004036  074256                                  EHT1
004040  074406                                  EDT1
004042  074476                                  EFT1

454
455                                     ;ERROR  226     "SSEI" SHOULD BE SET AFTER DATA TRANSFER
456
004044  072606                                  EMT226
004046  074256                                  EHT1
004050  074406                                  EDT1
004052  074476                                  EFT1

457
458                                     ;ERROR  227     SSE WAS DETECTED W/ SSEI SET
459
004054  072616                                  EMT227
004056  074256                                  EHT1
004060  074406                                  EDT1
004062  074476                                  EFT1

460
461                                     ;ERROR  230     WRONG UNIT SELECTED
462
004064  072632                                  EMT230
004066  074310                                  EHT112
004070  074426                                  EDT111
004072  074516                                  EFT111

463
464                                     ;ERROR  231     INCORRECT "OPE" STATUS
465
004074  072636                                  EMT231
004076  074310                                  EHT112
004100  074426                                  EDT111
004102  074516                                  EFT111

466
467                                     ;ERROR  232     UNUSED
468
004104  072646                                  EMT232
004106  000000                                  0
004110  000000                                  0
004112  000000                                  0

469
```

ERROR POINTER TABLE

```
470                                     ;ERROR  233     UNUSED
471
     004114  072650                               EMT233
     004116  000000                               0
     004120  000000                               0
     004122  000000                               0

472
473                                     ;ERROR  234     UNUSED
474
     004124  000000                               0
     004126  000000                               0
     004130  000000                               0
     004132  000000                               0

475
476                                     ;ERROR  235     UNUSED
477
     004134  000000                               0
     004136  000000                               0
     004140  000000                               0
     004142  000000                               0

478
479                                     ;ERROR  236     UNUSED
480
     004144  000000                               0
     004146  000000                               0
     004150  000000                               0
     004152  000000                               0

481
482                                     ;ERROR  237     UNUSED
483
     004154  000000                               0
     004156  000000                               0
     004160  000000                               0
     004162  000000                               0

484
485                                     ;ERROR  240     UNUSED
486
     004164  000000                               0
     004166  000000                               0
     004170  000000                               0
     004172  000000                               0

487
488                                     ;ERROR  241     UNUSED
489
     004174  000000                               0
     004176  000000                               0
     004200  000000                               0
     004202  000000                               0

490
491                                     ;ERROR  242     UNUSED
```

ERROR POINTER TABLE

```
492
     004204  000000                          0
     004206  000000                          0
     004210  000000                          0
     004212  000000                          0

493
494                                ;ERROR  243     UNUSED
495
     004214  000000                          0
     004216  000000                          0
     004220  000000                          0
     004222  000000                          0

496
497                                ;ERROR  244     UNUSED
498
     004224  000000                          0
     004226  000000                          0
     004230  000000                          0
     004232  000000                          0

499
500                                ;ERROR  245     UNUSED
501
     004234  000000                          0
     004236  000000                          0
     004240  000000                          0
     004242  000000                          0

502
503                                ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
504
     004244  072676                          EMT246
     004246  074256                          EHT1
     004250  074406                          EDT1
     004252  074476                          EFT1

505
506                                ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
507
     004254  072706                          EMT247
     004256  074256                          EHT1
     004260  074406                          EDT1
     004262  074476                          EFT1

508
509                                ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
510
     004264  072720                          EMT250
     004266  074256                          EHT1
     004270  074406                          EDT1
     004272  074476                          EFT1

511
512                                ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
513
```

ERROR POINTER TABLE

```
004274  072734                          EMT251
004276  074270                          EHT2
004300  074416                          EDT2
004302  074506                          EFT2

514
515                             ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
516
004304  072742                          EMT252
004306  074270                          EHT2
004310  074416                          EDT2
004312  074506                          EFT2

517
518                             ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
519
004314  072750                          EMT253
004316  074256                          EHT1
004320  074406                          EDT1
004322  074476                          EFT1

520
521                             ;ERROR  254     INCORRECT "ILF" STATUS
522
004324  072766                          EMT254
004326  074256                          EHT1
004330  074406                          EDT1
004332  074476                          EFT1

523
524                             ;ERROR  255     INCORRECT "ATA" STATUS
525
004334  072776                          EMT255
004336  074256                          EHT1
004340  074406                          EDT1
004342  074476                          EFT1

526
527                             ;ERROR  256     INCORRECT "ILR" STATUS
528
004344  073006                          EMT256
004346  074332                          EHT256
004350  074432                          EDT223
004352  074522                          EFT223

529
530                             ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
531
004354  073016                          EMT257
004356  074256                          EHT1
004360  074406                          EDT1
004362  074476                          EFT1

532
533                             ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
534
004364  073030                          EMT260
```

ERROR POINTER TABLE

```
004366  074256                          EHT1
004370  074406                          EDT1
004372  074476                          EFT1

535
536                             ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
537
004374  073042                          EMT261
004376  074256                          EHT1
004400  074406                          EDT1
004402  074476                          EFT1

538
539                             ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
540
004404  073062                          EMT262
004406  074256                          EHT1
004410  074406                          EDT1
004412  074476                          EFT1

541
542                             ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
543
004414  073072                          EMT263
004416  074256                          EHT1
004420  074406                          EDT1
004422  074476                          EFT1

544
545                             ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
546
004424  073102                          EMT264
004426  074256                          EHT1
004430  074406                          EDT1
004432  074476                          EFT1

547
548                             ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
549
004434  073122                          EMT265
004436  074256                          EHT1
004440  074406                          EDT1
004442  074476                          EFT1

550
551                             ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
552
004444  073142                          EMT266
004446  074256                          EHT1
004450  074406                          EDT1
004452  074476                          EFT1

553
554                             ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
555                             ;               , ADDRESS IS TOO LARGE
556
004454  073154                          EMT267
```

ERROR POINTER TABLE

```
004456  074256                          EHT1
004460  074406                          EDT1
004462  074476                          EFT1

557
558                             ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
559
004464  073172                          EMT270
004466  074256                          EHT1
004470  074406                          EDT1
004472  074476                          EFT1

560
561                             ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
562                             ;               DIDN'T DROP
563
004474  073206                          EMT271
004476  074256                          EHT1
004500  074406                          EDT1
004502  074476                          EFT1

564
565                             ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
566
004504  073224                          EMT272
004506  074256                          EHT1
004510  074406                          EDT1
004512  074476                          EFT1

567
568                             ;ERROR  273     PIP STIL SET AFTER SEARCH
569
004514  073242                          EMT273
004516  074256                          EHT1
004520  074406                          EDT1
004522  074476                          EFT1

570
571                             ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
572                             ;               REGISTERS BUT MXF DID NOT SET
573
004524  073260                          EMT274
004526  074256                          EHT1
004530  074406                          EDT1
004532  074476                          EFT1

574
575                             ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
576                             ;               COMMAND STARTED
577
004534  073276                          EMT275
004536  074256                          EHT1
004540  074406                          EDT1
004542  074476                          EFT1

578
579                             ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
```

ERROR POINTER TABLE

```
                                        ;               ZERO
004544  073310                                  EMT276
004546  074256                                  EHT1
004550  074406                                  EDT1
004552  074476                                  EFT1

                                        ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
                                        ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
                                        ;               3) RUN TIMED OUT
004554  073324                                  EMT277
004556  074256                                  EHT1
004560  074406                                  EDT1
004562  074476                                  EFT1

                                        ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
                                        ;               WAS NOT VALID
004564  073342                                  EMT300
004566  074256                                  EHT1
004570  074406                                  EDT1
004572  074476                                  EFT1

                                        ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
                                        ;               IS VALID
004574  073362                                  EMT301
004576  074256                                  EHT1
004600  074406                                  EDT1
004602  074476                                  EFT1

                                        ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
004604  073404                                  EMT302
004606  074256                                  EHT1
004610  074406                                  EDT1
004612  074476                                  EFT1

                                        ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
004614  073416                                  EMT303
004616  074256                                  EHT1
004620  074406                                  EDT1
004622  074476                                  EFT1

                                        ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
004624  073430                                  EMT304
004626  074256                                  EHT1
004630  074406                                  EDT1
```

ERROR POINTER TABLE

```
        004632  074476                          EFT1

604
605                                     ;ERROR  305     INCORRECT ''IAE'' STATUS DURING DATA TRANSFER
606
        004634  073442                          EMT305
        004636  074256                          EHT1
        004640  074406                          EDT1
        004642  074476                          EFT1

607
608                                     ;ERROR  306     ''SKI'' ERROR DURING DATA TRANSFER
609
        004644  073454                          EMT306
        004646  074256                          EHT1
        004650  074406                          EDT1
        004652  074476                          EFT1

610
611                                     ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
612
        004654  073464                          EMT307
        004656  074256                          EHT1
        004660  074406                          EDT1
        004662  074476                          EFT1

613
614                                     ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
615
        004664  073504                          EMT310
        004666  074256                          EHT1
        004670  074406                          EDT1
        004672  074476                          EFT1

616
617                                     ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
618
        004674  073516                          EMT311
        004676  074256                          EHT1
        004700  074406                          EDT1
        004702  074476                          EFT1

619
620                                     ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
621
        004704  073530                          EMT312
        004706  074256                          EHT1
        004710  074406                          EDT1
        004712  074476                          EFT1

622
623                                     ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
624
        004714  073542                          EMT313
        004716  074256                          EHT1
        004720  074406                          EDT1
        004722  074476                          EFT1
```

```
625
626                                     ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
627
     004724  073562                             EMT314
     004726  074256                             EHT1
     004730  074406                             EDT1
     004732  074476                             EFT1

628
629                                     ;ERROR  315     WRITE LOCK ERROR
630
     004734  073574                             EMT315
     004736  074256                             EHT1
     004740  074406                             EDT1
     004742  074476                             EFT1

631
632                                     ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
633
     004744  073606                             EMT316
     004746  074256                             EHT1
     004750  074406                             EDT1
     004752  074476                             EFT1

634
635                                     ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
636
     004754  073620                             EMT317
     004756  074256                             EHT1
     004760  074406                             EDT1
     004762  074476                             EFT1

637
638                                     ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
639
     004764  073630                             EMT320
     004766  074256                             EHT1
     004770  074406                             EDT1
     004772  074476                             EFT1

640
641                                     ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
642
     004774  073640                             EMT321
     004776  074256                             EHT1
     005000  074406                             EDT1
     005002  074476                             EFT1

643
644                                     ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
645
     005004  073650                             EMT322
     005006  074256                             EHT1
     005010  074406                             EDT1
     005012  074476                             EFT1
```

```
646
647                                     ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
648
    005014  073656                              EMT323
    005016  074256                              EHT1
    005020  074406                              EDT1
    005022  074476                              EFT1

649
650                                     ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
651
    005024  073666                              EMT324
    005026  074256                              EHT1
    005030  074406                              EDT1
    005032  074476                              EFT1

652
653                                     ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
654
    005034  073700                              EMT325
    005036  074256                              EHT1
    005040  074406                              EDT1
    005042  074476                              EFT1

655
656                                     ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
657
    005044  073712                              EMT326
    005046  074256                              EHT1
    005050  074406                              EDT1
    005052  074476                              EFT1

658
659                                     ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
660
    005054  073730                              EMT327
    005056  074256                              EHT1
    005060  074406                              EDT1
    005062  074476                              EFT1

661
662                                     ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
663
    005064  073742                              EMT330
    005066  074256                              EHT1
    005070  074406                              EDT1
    005072  074476                              EFT1

664
665                                     ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
666
    005074  073752                              EMT331
    005076  074256                              EHT1
    005100  074406                              EDT1
    005102  074476                              EFT1

667
```

ERROR POINTER TABLE

```
668                                     ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
669
    005104  073764                              EMT332
    005106  074256                              EHT1
    005110  074406                              EDT1
    005112  074476                              EFT1

670
671                                     ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
672
    005114  073776                              EMT333
    005116  074256                              EHT1
    005120  074406                              EDT1
    005122  074476                              EFT1

673
674                                     ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
675
    005124  074014                              EMT334
    005126  074256                              EHT1
    005130  074406                              EDT1
    005132  074476                              EFT1

676
677                                     ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
678
    005134  074032                              EMT335
    005136  074256                              EHT1
    005140  074406                              EDT1
    005142  074476                              EFT1

679
680                                     ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
681
    005144  074052                              EMT336
    005146  074344                              EHT336
    005150  074442                              EDT336
    005152  074532                              EFT336

682
683                                     ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
684
    005154  074062                              EMT337
    005156  074356                              EHT337
    005160  074452                              EDT337
    005162  074542                              EFT337

685
686                                     ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
687
    005164  074072                              EMT340
    005166  074344                              EHT336
    005170  074442                              EDT336
    005172  074532                              EFT336

688
689                                     ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
```

ERROR POINTER TABLE

```
690
     005174  074104                          EMT341
     005176  074344                          EHT336
     005200  074442                          EDT336
     005202  074532                          EFT336

691
692                                 ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
693
     005204  074112                          EMT342
     005206  074256                          EHT1
     005210  074406                          EDT1
     005212  074476                          EFT1

694
695                                 ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
696
     005214  074124                          EMT343
     005216  074256                          EHT1
     005220  074406                          EDT1
     005222  074476                          EFT1

697
698                                 ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
699
     005224  074136                          EMT344
     005226  074370                          EHT344
     005230  074462                          EDT344
     005232  074552                          EFT344

700
701                                 ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
702
     005234  074150                          EMT345
     005236  074256                          EHT1
     005240  074406                          EDT1
     005242  074476                          EFT1

703
704                                 ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
705
     005244  074160                          EMT346
     005246  074256                          EHT1
     005250  074406                          EDT1
     005252  074476                          EFT1

706
707                                 ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
708
     005254  074174                          EMT347
     005256  074256                          EHT1
     005260  074406                          EDT1
     005262  074476                          EFT1

709
710                                 ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
711
```

ERROR POINTER TABLE

```
005264  074206                          EMT350
005266  074256                          EHT1
005270  074406                          EDT1
005272  074476                          EFT1

712
713                             ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
714
005274  074224                          EMT351
005276  074256                          EHT1
005300  074406                          EDT1
005302  074476                          EFT1

715
716                             ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
717
005304  074234                          EMT352
005306  000000                          0
005310  000000                          0
005312  000000                          0

718
719                             ;ERROR  353     LOOK AHEAD TEST FAILS
720
005314  074240                          EMT353
005316  074402                          EHT353
005320  074474                          EDT353
005322  074562                          EFT353

721
722                             ;ERROR  354     BSE SHOULD NOT BE SET
723
005324  074250                          EMT354
005326  074256                          EHT1
005330  074406                          EDT1
005332  074476                          EFT1

724
725                             ;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
                        ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
                        ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
                        ;NO MESSAGE TO BE TYPED FOR THE ERROR.
                        ;
                        ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
                        ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
                        ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
                        ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
                        ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
                        ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
                        ;MUST ALSO HAVE A FORMAT.
                        ;
                        ;IN SUMMARY,
                        ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
                        ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
                        ;       OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
```

ERROR TABLE USAGE

```
                                        ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS

005334  011600                  BADTMO: MOV     (SP),R0         ;SAVE PC WHERE THE TIME OUT OCCURED
005336  005740                          TST     -(R0)           ;ADJUST PC -2
005340  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
005342  104401  005350                  TYPE    ,65$            ;;TYPE ASCIZ STRING
005346  000417                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
005406                          64$:
005406  010046                          MOV     R0,-(SP)        ;SETUP FOR TYPING OUT PC
005410  104402                          TYPOC
005412  000240                          NOP                     ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
                                                                ;TO STOP ON UNEXPECTED TIMEOUT.
005414  000404                          BR      START           ;BRANCH TO START

                                        .SBTTL  START OF PROGRAM

005416  012737  177777  001326  START1: MOV     #-1,CHGADR      ;CHANGE RH/RM BUS ADDRESS
005424  000402                          BR      START2

005426  005037  001326          START:  CLR     CHGADR          ;NO CHANGE IN ADDRESS
005432  000240                  START2: NOP
005434  005227  000000                  INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
005440  001375                          BNE     .-4             ;OF WORD
005442  000005                          RESET                   ;RESET THE WORLD

                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
005444  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
005450  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
005452  022706  001154                  CMP     #SWR,R6         ;;DONE?
005456  001374                          BNE     .-6             ;;LOOP BACK IF NO
005460  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
005464  012737  062072  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005472  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005500  012737  062762  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005506  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005514  012737  065766  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005522  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
005530  012737  066074  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
005536  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
005544  013737  041350  041342          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
005552  005037  001206                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
005556  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
005562  112737  000001  001131          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
005570  012737  005570  001122          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005576  012737  005576  001124          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
005604  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005610  012737  005644  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
005616  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005624  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005632  022777  177777  173314          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005640  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
```

INITIALIZE THE COMMON TAGS

```
005642  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
005644  012716  005652          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
005650  000002                          RTI
005652  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005660  012737  000174  001156          MOV     #DISPREG,DISPLAY
005666  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

005672  005037  001230                  CLR     $PASS           ;;CLEAR PASS COUNT
005676  132737  000200  001243          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005704  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
005706  012737  001244  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005714                          67$:
 25                             ;SETUP "TIMEOUT" TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
 26 005714  012737  005334  000004      MOV     #BADTMO,ERRVEC  ;SETUP FOR UNEXPECTED TIMEOUT
 27 005722  012737  000300  000006      MOV     #PR6,ERRVEC+2   ;LEVEL 6
 28
 29                             .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005730  005227  177777                  INC     #-1             ;;FIRST TIME?
005734  001033                          BNE     68$             ;;BRANCH IF NO
005736  022737  041512  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
005744  001427                          BEQ     68$             ;;BRANCH IF YES
005746  104401  005754                  TYPE    ,69$            ;;TYPE ASCIZ STRING
005752  000424                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF>@CZRNDA0 - RM80 FUNCTIONAL TEST, PT 1@<CRLF>
006024                          68$:
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
006024  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
006030  001012                          BNE     70$             ;;BRANCH IF YES
006032  123727  001242  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
006040  001406                          BEQ     70$             ;;BRANCH IF YES
006042  023727  001154  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
006050  001005                          BNE     71$             ;;BRANCH IF NO
006052  104407                          GTSWR                   ;;GET SOFT-SWR SETTINGS
006054  000403                          BR      71$
006056  112737  000001  001150  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
006064                          71$:
 30
 31                             ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
 32                             ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
 33
 34 006064  005037  001332              CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
 35 006070  122737  000016  000041      CMPB    #16,@#41        ;LOADED FROM AN RM80 ?
 36 006076  001121                      BNE     3$              ;BR IF NOT
 37 006100  013737  000040  001332      MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
 38 006106  122737  000007  001332      CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
 39 006114  103002                      BHIS    1$              ;YES
 40 006116  105037  001332              CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
 41 006122  005737  000042      1$:     TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
 42 006126  001425                      BEQ     2$              ;BR IF NEITHER
 43 006130  104401  006136              TYPE    ,73$            ;;TYPE ASCIZ STRING
    006134  000412                      BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
006162                          72$:
 44 006162  005046                      CLR     -(SP)           ;CLEAR WORD ON STACK
 45 006164  113716  001332              MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
 46 006170  104403                      TYPOS                   ;TYPE THE ADDRESS
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
 47 006172   001                          .BYTE   1               ;ONLY 1 CHARACTER
 48 006173   000                          .BYTE   0               ;SUPRESS LEADING ZEROS
 49 006174   104401  001217               TYPE    ,$CRLF          ;CR-LF
 50 006200   000460                       BR      3$              ;GET NUMBER OF DRIVES
 51
 52 006202   005227  177777     2$:       INC     #-1             ;FIRST TIME THRU HERE ?
 53 006206   001055                       BNE     3$              ;NO
 54 006210   104401  006216               TYPE    ,75$            ;;TYPE ASCIZ STRING
    006214   000410                       BR      74$             ;;GET OVER THE ASCIZ
                                 ;;75$:    .ASCIZ  <CRLF>/TO TEST DRIVE /
    006236                       74$:
 55 006236   005046                       CLR     -(SP)           ;CLEAR WORD ON STACK
 56 006240   113716  001332               MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
 57 006244   104403                       TYPOS                   ;TYPE DRIVE ADDRESS
 58 006246      001                       .BYTE   1               ;ONLY 1 CHARACTER
 59 006247      000                       .BYTE   0               ;SUPRESS LEADING ZEROS
 60 006250   104401  006256               TYPE    ,76$            ;;TYPE ASCIZ STRING
    006254   000432                       BR      3$              ;;GET OVER THE ASCIZ
                                 ;;76$:    .ASCIZ  /, HALT PROGRAM, CLEAR LOC. 40 AND RESTART PROGRAM./<CRLF>
    006342                       3$:
 64                              ;CHECK FOR AUTO MODE OR STANDALONE MODE
 65 006342   005737  000042               TST     @#42            ;RUNNING IN AUTO MODE ?
 66 006346   001537                       BEQ     STANDALONE      ;BR IF NO
 67 006350   012737  000377  001300       MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
 68
 69                              ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
 70 006356                       XSIZ:
 71 006356   132737  000200  001243       BITB    #BIT7,$ENVM     ;SIZING ALLOWED ?
 72 006364   001124                       BNE     12$             ;NO
 73
 74 006366   005001                       CLR     R1              ;START FROM DRIVE 0
 75 006370   013700  001276               MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
 76 006374   104401  067112               TYPE    ,SYSTAT         ;TYPE 'UNIT STATUS:'
 77
 78 006400   136137  070102  001300  1$:  BITB    ATNTBL(R1),$DEVM        ;IS DEVICE PRESENT IN MAP ?
 79 006406   001507                       BEQ     11$             ;BR IF NO
 80 006410   104401  001217               TYPE    ,$CRLF          ;CR-LF
 81 006414   010146                       MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
    006416   104403                       TYPOS                   ;;GO TYPE--OCTAL ASCII
    006420      002                       .BYTE   2               ;;TYPE 2 DIGIT(S)
    006421      000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
 82 006422   104401  067775               TYPE    ,BLNKS4         ;TYPE 4 BLANKS
 83
 84 006426   012760  000040  000010       MOV     #CLR,RMCS2(R0)  ;CLEAR MASS BUS
 85 006434   010160  000010               MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
 86 006440   005760  000012               TST     RMDS(R0)        ;ACCESS DRIVE REGISTER
 87 006444   032760  010000  000010       BIT     #NED,RMCS2(R0)  ;IS DRIVE PRESENT ?
 88 006452   001027                       BNE     3$              ;BR IF NO
 89 006454   032760  004000  000000       BIT     #DVA,RMCS1(R0)  ;IS DRIVE AVAILABLE ?
 90 006462   001426                       BEQ     4$              ;BR IF NO
 91 006464   012737  067130  006624       MOV     #$RM80,10$      ;ASSUME RM80 DEVICE
 92 006472   022760  020026  000026       CMP     #20026,RMDT(R0) ;SINGLE PORT RM80 ?
 93 006500   001407                       BEQ     2$              ;BR IF YES
 94 006502   022760  024026  000026       CMP     #24026,RMDT(R0) ;DUAL PORT RM80 ?
 95 006510   001403                       BEQ     2$              ;BR IF YES
 96 006512   104401  067135               TYPE    ,NOTRM          ;DRIVE NOT AN RM80
 97 006516   000412                       BR      5$              ;CHECK NEXT DRIVE
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
 98 006520  032760  010000  000012  2$:     BIT     #MOL,RMDS(R0)   ;IS MEDIUM ON LINE ?
 99 006526  001412                          BEQ     6$              ;BR IF NO
100 006530  000417                          BR      7$
101
102 006532  104401  067167          3$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
103 006536  000402                          BR      5$              ;CHECK NEXT DRIVE
104
105 006540  104401  067204          4$:     TYPE    ,NOTAVL         ;DRIVE NOT AVAILABLE
106 006544  146137  070102  001300  5$:     BICB    ATNTBL(R1),$DEVM        ;CLEAR DEVICE FROM BIT MAP
107 006552  000425                          BR      11$             ;CHECK NEXT DRIVE
108
109 006554  104401  067223          6$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
110 006560  146137  070102  001300          BICB    ATNTBL(R1),$DEVM        ;CLEAR DEVICE FROM BIT MAP
111 006566  000413                          BR      9$              ;PRINT DRIVE TYPE
112
113 006570  005737  001332          7$:     TST     XXDP            ;LOADED FROM RM80 ?
114 006574  001406                          BEQ     8$              ;NO
115 006576  123701  001332                  CMPB    XXDP,R1         ;IS THIS THE DRIVE ?
116 006602  001003                          BNE     8$              ;BR IF NO
117 006604  104401  067152                  TYPE    ,LODEV          ;DRIVE IS LOAD DEVICE
118 006610  000755                          BR      5$
119
120 006612  104401  067234          8$:     TYPE    ,UNTON          ;DRIVE ONLINE
121 006616  104401  067777          9$:     TYPE    ,BLNKS2         ;TYPE 2 BLANKS
122 006622  104401                          TYPE                    ;PRINT DRIVE TYPE
123 006624  000000                  10$:    .WORD   0               ;MESSAGE ADDRESS HERE
124
125 006626  005201                  11$:    INC     R1              ;INCREMENT THE DRIVE ADDRESS
126 006630  020127  000007                  CMP     R1,#7           ;ALL DRIVES ARE CHECKED ?
127 006634  003661                          BLE     1$              ;BRANCH IF NOT
128
129 006636  104401  001217          12$:    TYPE    ,$CRLF          ;CR-LF
130 006642  000137  007402                  JMP     CMNSTART        ;JUMP TO COMMON START
```

```
                                .SBTTL  STANDALONE INPUT ROUTINES

006646                          STANDALONE:
006646  004737  064402                JSR     PC,$TKINT       ;INITIALIZE CONSOLE

006652  005227  177777                INC     #-1             ;FIRST TIME THRU HERE ?
006656  001023                        BNE     2$              ;BR IF NO

                                ;SEE IF OPERATOR WANTS HELP TEXT

006660  104401  066536                TYPE    ,MSHELP         ;WANT HELP ?
006664  104411                        RDCHR                   ;GET RESPONSE
006666  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006672  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ?
006700  001005                        BNE     1$              ;NO
006702  104401  001176                TYPE    ,$TMP1          ;TYPE 'Y'
006706  104401  104372                TYPE    ,HELP           ;YES - TYPE HELP TEXT
006712  000414                        BR      3$
006714  104401  067771        1$:     TYPE    ,N              ;TYPE 'N'
006720  104401  001217                TYPE    ,$CRLF          ;CR-LF
006724  000407                        BR      3$

                                ;SEE IF USER WANTS TO CHANGE UNIBUS ADDRESS
006726                          2$:
006726  005737  001326                TST     CHGADR          ;CHANGE RH/RM BUS ADDRESS ?
006732  001457                        BEQ     7$              ;BR IF NO
006734  005037  001326                CLR     CHGADR          ;NO CHANGE NEXT TIME
006740  104401  001217                TYPE    ,$CRLF          ;CR-LF

                                ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY
006744                          3$:
006744  104401  066567                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
006750  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
006754  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006756  104401  067777                TYPE    ,BLNKS2         ;TYPE 2 BLANKS
006762  104413                        RDOCT                   ;GET NEW BUS ADDRESS
006764  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN ?
006770  001412                        BEQ     5$              ;YES-SKIP TO NEXT ENTRY
006772  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE ?
007000  101403                        BLOS    4$              ;YES
007002  104401  066577                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
007006  000756                        BR      3$              ;TRY AGAIN
007010  013737  001176  001276  4$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS

007016  104401  066641          5$:   TYPE    ,CNSL03
007022  005046                        CLR     -(SP)
007024  113716  001272                MOVB    $VECT1,(SP)     ;GET CURRENT VECTOR ADDRESS
007030  104402                        TYPOC
007032  104401  067777                TYPE    ,BLNKS2         ;TYPE 2 BLANKS
007036  104413                        RDOCT                   ;GET NEW VECTOR ADDRESS
007040  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
007044  001412                        BEQ     7$              ;YES-SKIP TO NEXT ENTRY
007046  022737  001000  001176        CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000 ?
007054  101003                        BHI     6$              ;YES!!
007056  104401  066650                TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
007062  000755                        BR      5$              ;RETRY
007064  113737  001176  001272  6$:   MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
```

STANDALONE INPUT ROUTINES

```
                                        ;FIND OUT IF USER WANTS TO DO LOGICAL ADDRESS PLUG TEST
007072  005037  001330          7$:     CLR     MANUAL          ;NO MANUAL INTERVETION
007076  104401  067300                  TYPE    ,MSMNUL         ;TYPE 'PLUG TEST ?'
007102  104412                          RDLIN                   ;READ THE ENTRY
007104  012600                          MOV     (SP)+,R0        ;SAVE ADDRESS OF RESPONSE
007106  105710                          TSTB    (R0)            ;WAS RESPONSE A CARRIAGE RETURN ?
007110  001417                          BEQ     9$              ;BR IF YES
007112  105760  000001                  TSTB    1(R0)           ;WAS IT TERMINATED WITH CARRIAGE RETURN ?
007116  001011                          BNE     8$              ;BR IF NO
007120  122710  000116                  CMPB    #'N,(R0)        ;WAS IT A 'N' RESPONSE ?
007124  001411                          BEQ     9$              ;BR IF YES
007126  012737  000001  001330          MOV     #1,MANUAL       ;SET MANUAL INTERVENTION FLAG
007134  122710  000131                  CMPB    #'Y,(R0)        ;WAS IT A 'Y' RESPONSE ?
007140  001403                          BEQ     9$              ;BR IF YES
007142  104401  067047          8$:     TYPE    ,CNSL09         ;TYPE '' ?ILLEGAL INPUT''
007146  000751                          BR      7$              ;RETRY

                                        ;DIALOGUE TO INPUT DEVICE NUMBERS
007150  005227  177777          9$:     INC     #-1             ;FIRST TIME THRU ?
007154  001002                          BNE     10$             ;BR IF NO
007156  104401  066704                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
007162  104401  001217          10$:    TYPE    ,$CRLF          ;CR-LF
007166  005037  001300          11$:    CLR     $DEVM           ;CLEAR DEVICE MAP
007172  104401  067070                  TYPE    ,MSDRVS         ;TYPE 'DRIVE(S): '
007176  104411                          RDCHR
007200  012637  001176                  MOV     (SP)+,$TMP1     ;GET RESPONSE
007204  023727  001176  000101          CMP     $TMP1,#'A       ;IS INPUT ''A'' ?
007212  001007                          BNE     12$             ;NO
007214  104401  066522                  TYPE    ,ALL            ;YES, TYPE ''ALL'' AND GO
007220  012737  000377  001300          MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
007226  000137  006356                  JMP     XSIZ            ;AUTO SIZE.

007232  023727  001176  000015  12$:    CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
007240  001436                          BEQ     14$             ;YES
007242  104401  001176                  TYPE    ,$TMP1          ;ECHO RESPONSE
007246  023727  001176  000060          CMP     $TMP1,#'0       ;NUMBER < 0 ?
007254  002430                          BLT     14$             ;YES
007256  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7 ?
007264  003427                          BLE     15$             ;NO
007266  000423                          BR      14$             ;ILLEGAL INPUT

007270  104411          13$:    RDCHR
007272  012637  001176                  MOV     (SP)+,$TMP1     ;GET RESPONSE
007276  023727  001176  000015          CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
007304  001432                          BEQ     16$             ;YES
007306  104401  066533                  TYPE    ,COMMA          ;TYPE ',  '
007312  104401  001176                  TYPE    ,$TMP1          ;ECHO RESPONSE
007316  023727  001176  000060          CMP     $TMP1,#'0       ;NUMBER < 0 ?
007324  002404                          BLT     14$             ;YES
007326  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7 ?
007334  003403                          BLE     15$             ;NO
007336  104401  067046          14$:    TYPE    ,CNSL08         ;TYPE CR-LF '' ?ILLEGAL INPUT''
007342  000711                          BR      11$             ;RETRY

007344  013701  001176          15$:    MOV     $TMP1,R1        ;R1 = DRIVE NUMBER
007350  042701  177770                  BIC     #^C7,R1
```

```
    114 007354  156137  070102  001300          BISB    ATNTBL(R1),$DEVM        ;SET DEVICE IN MAP
    115 007362  122737  000377  001300          CMPB    #377,$DEVM      ;DONE ?
    116 007370  101337                          BHI     13$             ;NO
    117 007372  104401  001217          16$:    TYPE    ,$CRLF          ;CR-LF
    118 007376  000137  006356                  JMP     XSIZ            ;GO SIZE DEVICES
```

```
                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
007402                                  CMNSTART:
007402  104401  067244                          TYPE    ,DRIVES         ;TYPE 'DRIVE(S) TO BE TESTED'
007406  013700  001300                          MOV     $DEVM,R0        ;R0 = DEVICE MAP
007412  001004                                  BNE     1$              ;BR IF DRIVES TO TEST
007414  104401  066533                          TYPE    ,COMMA          ;TYPE ', '
007420  104401  067273                          TYPE    ,NONE           ;TYPE 'NONE'
007424  012701  001470                  1$:     MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
007430  010137  001466                          MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
007434  012702  000001                          MOV     #1,R2           ;R2 = DEVICE POINTER
007440  005003                                  CLR     R3              ;R3 = DEVICE NUMBER
007442  030200                          2$:     BIT     R2,R0           ;IS THIS DEVICE IN MAP ?
007444  001413                                  BEQ     3$              ;NO !!
007446  104401  066533                          TYPE    ,COMMA          ;TYPE ', '
007452  010311                                  MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
007454  010346                                  MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
007456  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
007460     001                                  .BYTE   1               ;;TYPE 1 DIGIT(S)
007461     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
007462  116361  070102  000001                  MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
007470  062701  000002                          ADD     #2,R1           ;ADVANCE ENTRY POINTER
007474  006302                          3$:     ASL     R2              ;ADVANCE DEVICE POINTER
007476  105702                                  TSTB    R2              ;DONE ALL DEVICES ?
007500  001402                                  BEQ     4$              ;YES
007502  005203                                  INC     R3              ;ADVANCE DEVICE NUMBER
007504  000756                                  BR      2$              ;ENTER NEXT DEVICE
007506  005011                          4$:     CLR     (R1)            ;TERMINATE TEST QUE
007510  104401  001217                          TYPE    ,$CRLF          ;TYPE CR-LF

                                        ;SIZE FOR CLOCK
007514  004737  043246                          JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
007520  000425                                  BR      6$              ;YES - CLOCK IS PRESENT
007522  104401  007530                          TYPE    ,65$            ;;TYPE ASCIZ STRING
007526  000413                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/NO 'L' OR 'P' CLOCK/
007556                                  64$:
007556  005737  000042                          TST     @#42            ;ANY MONITOR PRESENT ?
007562  001002                                  BNE     5$              ;BR IF YES
007564  000137  005426                          JMP     START           ;JUMP TO START
007570  000137  041502                  5$:     JMP     $GET42          ;RETURN CONTROL TO MONITOR
007574                                  6$:

007574  000240                          READY:  NOP                     ;READY TO START TEST
007576  105737  001300                          TSTB    $DEVM           ;ANY DRIVES IN MAP ?
007602  001007                                  BNE     2$              ;BR IF YES
007604  005737  000042                          TST     @#42            ;ANY MONITOR PRESENT ?
007610  001002                                  BNE     1$              ;BR IF YES
007612  000137  005426                          JMP     START           ;JUMP TO START
007616  000137  041502                  1$:     JMP     $GET42          ;RETURN CONTROL TO MONITOR

007622  105037  001116                  2$:     CLRB    $TSTNM          ;RESET TEST NUMBER
007626  005037  001206                          CLR     $TIMES          ;INITIALIZE NUMBER OF ITERATIONS
007632  004737  064402                          JSR     PC,$TKINT       ;INITIALIZE TTY
007636  012746  000240                          MOV     #PR5,-(SP)      ;;PUT NEW PS ON STACK
007642  012746  007650                          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
007646  000002                                  RTI                     ;;POP NEW PC AND PS
007650                                  64$:
```

STANDALONE INPUT ROUTINES

```
     49 007650  117737  171612  001234          MOVB    @TSTQUE,$UNIT   ;LOAD DRIVE NUMBER
     50
     51                                  ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND SET LAST TRACK ADDRESS
     52 007656  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     53 007662  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
     54 007670  113760  001234  000010          MOVB    $UNIT,RMCS2(R0) ;SELECT DEVICE UNDER TEST
     55 007676  012737  006400  001334          MOV     #TAB!TA4!TA1,LSTRK      ;SET LAST TRACK = 13.
     56
     57                                  ;TYPE DRIVE NUMBER TO BE TESTED($UNIT)
     58 007704  104401  001217                  TYPE    ,$CRLF          ;CR-LF
     59 007710  104401  067104                  TYPE    ,MSGDRV         ;TYPE 'DRIVE'
     60 007714  013746  001234                  MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
        007720  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
        007722     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
        007723     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
     61 007724  005004                          CLR     R4              ;THESE TWO LOOPS ARE ADDED TO
     62 007726  005304                          DEC     R4              ;WAIT FOR TTY
     63 007730  001376                          BNE     .-2
     64 007732  005304                          DEC     R4
     65 007734  001376                          BNE     .-2
```

```
                        ;;*****************************************************************
                        ;*TEST 1         CONTROLLER ACCESS TEST
                        ;;*****************************************************************
                        TST1:
007736                          SCOPE                   ;SCOPE CALL
007736  000004                  NOP                     ;START OF TEST
007740  000240
007742  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
007746  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
007752  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
007756  012737  000001  001226  MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
007764  005001                  CLR     R1
007766  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
007772  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
007776  012737  010100  000004  MOV     #1$,ERRVEC
010004  012737  000300  000006  MOV     #PR6,ERRVEC+2
010012  110160  000001          MOVB    R1,RMCS1+1(R0)  ;MOVE HI BYTE TO RMCS1
010016  010160  000002          MOV     R1,RMWC(R0)     ;MOVE WORD COUNT REGISTER
010022  016002  000002          MOV     RMWC(R0),R2
010026  010160  000004          MOV     R1,RMBA(R0)     ;MOVE BUS ADDRESS REGISTER
010032  016002  000004          MOV     RMBA(R0),R2
010036  016046  000010          MOV     RMCS2(R0),-(SP)         ;;PUSH RMCS2(R0) ON STACK
010042  010160  000010          MOV     R1,RMCS2(R0)    ;MOVE CONTROL STATUS REGISTER
010046  016002  000010          MOV     RMCS2(R0),R2
010052  012660  000010          MOV     (SP)+,RMCS2(R0)         ;;POP STACK INTO RMCS2(R0)
010056  010160  000022          MOV     R1,RMDB(R0)     ;MOVE DATA BUFFER
010062  016002  000022          MOV     RMDB(R0),R2
010066  012637  000006          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
010072  012637  000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
010076  000417                  BR      3$              ;NO BUS TIMEOUT OCCURRED

010100  022626          1$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
010102  012637  000006          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
010106  012637  000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
010112  104110                  EMT     110
010114  005737  000042          TST     @#42            ;IS MONITOR PRESENT ?
010120  001002                  BNE     2$              ;BR IF YES
010122  000137  005416          JMP     START1          ;YES-GO GET $BASE
010126  005037  001300  2$:     CLR     $DEVM           ;FUDGE NO DRIVES IN MAP
010132  000137  041502          JMP     $GET42          ;RETURN CONTROL TO MONITOR
010136                  3$:

                        ;;*****************************************************************
                        ;*TEST 2         LOGICAL ADDRESS PLUGS TEST
                        ;;*****************************************************************
                        TST2:
010136                          SCOPE                   ;SCOPE CALL
010136  000004                  NOP                     ;START OF TEST
010140  000240
010142  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
010146  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
010152  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
010156  012737  000001  001206  MOV     #1,$TIMES       ;;DO 1 ITERATION
010164  012737  000002  001226  MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

010172  005737  001330          TST     MANUAL          ;DO WANT MANUAL INTERVENTION TEST ?
```

```
                                        BEQ     7$              ;NO--SKIP TEST

                                        MOV     #2$,$LPERR      ;SET ERROR LOOP ADDRESS
                                        TYPE    ,MSUNIT         ;TYPE UNIT MESSAGE
                                        MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                ;;GET DRIVE NUMBER BEING TESTED
010216  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
010220  001                             .BYTE   1               ;;TYPE 1 DIGIT(S)
010221  000                             .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TYPE    ,MINSTR         ;TYPE INSTRUCTIONS MESSAGE
                                        CLR     R2              ;START WITH UNIT #0
                                        TST     $UNIT           ;IS THIS UNIT #0 ?
                                        BNE     1$              ;NO
                                        TYPE    ,PULL           ;TYPE 'PULL LOGICAL ADRS PLUG #0'
                                        TYPE    ,$CRLF          ;CR-LF

                                1$:     TYPE    ,INSERT         ;TYPE 'INSERT LOGICAL ADRS PLUG #'
                                        MOV     R2,-(SP)        ;;SAVE R2 FOR TYPEOUT
                                                                ;;TYPE UNIT TO BE ADDRESSED
010254  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
010256  001                             .BYTE   1               ;;TYPE 1 DIGIT(S)
010257  000                             .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TYPE    ,RET            ;TYPE RETURN WHEN READY
                                        RDLIN                   ;WAIT FOR <RET> TO CONTINUE
                                        TYPE    ,$CRLF          ;CR-LF

                                        MOV     R2,RMCS2(R0)    ;ADDRESS UNIT NUMBER IN R2
                                        MOV     #OPE,$TMP0      ;GET EXPECTED STATUS
                                        MOV     RMER2(R0),$TMP1 ;GET RECEIVED STATUS
                                        BIC     #^C<OPE>,$TMP1  ;IS "OPE" SET ?
                                        BNE     2$              ;BR IF YES
                                        EMT     231

                                2$:     MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
                                        MOV     R2,RMCS2(R0)    ;SELECT DEVICE FOR TEST

                                        CLR     $TMP0           ;GET EXPECTED STATUS
                                        MOV     RMER2(R0),$TMP1 ;GET RECEIVED STATUS
                                        BIC     #^C<OPE>,$TMP1  ;IS "OPE" SET ?
                                        BEQ     3$              ;BR IF NO
                                        EMT     231

                                3$:     MOV     RMCS2(R0),$TMP1 ;GET RECEIVED/UNIT ADDRESS BITS
                                        BIC     #^CUNTMSK,$TMP1 ;MASK UNIT ADDRESS BITS
                                        MOV     R2,$TMP0        ;GET EXPECTED UNIT ADDRESS BITS
                                        CMP     $TMP1,R2        ;WAS CORRECT UNIT ADDRESSED ?
                                        BEQ     4$              ;BR IF YES
                                        EMT     230
                                        BR      6$

                                4$:     MOV     RMCS2(R0),$TMP0 ;GET RMCS2 STATUS
                                        MOV     RMCS1(R0),$TMP1 ;GET RMCS1 STATUS
                                        BIT     #NED,$TMP0      ;WAS DEVICE NON-EXISTENT ?
                                        BEQ     5$              ;BR IF NO
                                        EMT     111
                                        BR      6$
```

```
010444 032737 004000 001176 5$:    BIT    #DVA,$TMP1        ;WAS DEVICE AVAILABLE ?
010452 001001                      BNE    6$                ;BR IF YES
010454 104112                      EMT    112

010456 005202               6$:    INC    R2                ;ADDRESS NEXT UNIT
010460 020227 000007               CMP    R2,#UNTMSK        ;ALL PLUGS ADDRESSED ?
010464 101670                      BLOS   1$                ;BR IF NO
010466 104401 067750               TYPE   ,DONE             ;TYPE DONE MESSAGE
010472 104401 067617               TYPE   ,INSERT           ;TYPE 'INSERT LOGICAL ADRS PLUG #'
010476 013746 001234               MOV    $UNIT,-(SP)       ;;SAVE $UNIT FOR TYPEOUT
                                                            ;;GET ORIGINAL DRIVE ADDRESSED
010502 104403                      TYPOS                    ;;GO TYPE--OCTAL ASCII
010504    001                      .BYTE  1                 ;;TYPE 1 DIGIT(S)
010505    000                      .BYTE  0                 ;;SUPPRESS LEADING ZEROS
010506 104401 067720               TYPE   ,RET              ;TYPE RETURN WHEN READY
010512 104412                      RDLIN                    ;WAIT FOR <RET> TO CONTINUE
010514 104401 001217               TYPE   ,$CRLF            ;CR-LF
010520 005004                      CLR    R4                ;THESE TWO LOOPS ARE ADDED TO
010522 005304                      DEC    R4                ;WAIT FOR TTY
010524 001376                      BNE    .-2
010526 005304                      DEC    R4
010530 001376                      BNE    .-2
010532                      7$:


                            ;;*****************************************************************
                            ;;*TEST 3          UNIBUS INITIALIZE TEST
                            ;;*****************************************************************
010532                      TST3:
010532 000004                      SCOPE                    ;SCOPE CALL
010534 000240                      NOP                      ;START OF TEST
010536 012706 001100               MOV    #STACK,SP         ;INITIALIZE STACK POINTER
010542 013700 001276               MOV    $BASE,R0          ;R0 = UNIBUS ADDRESS
010546 013701 001466               MOV    TSTQUE,R1         ;(R1) = DEVICE BEING TESTED
010552 012737 000001 001206        MOV    #1,$TIMES         ;;DO 1 ITERATION
010560 012737 000003 001226        MOV    #3,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX

010566 004737 052466               JSR    PC,CNTCLR         ;GO ISSUE CONTROLLER CLEAR
010572 000404                      BR     1$                ;GO TO 1$ IF NO ERROR
010574 000240                      NOP                      ;RETURN HERE IF ERROR
010576 104000                      EMT                      ;ERROR NUMBER DEFINED BY SUBROUTINE
010600 000137 011402               JMP    15$               ;GO TO 15$ IF ERROR
010604                      1$:
010604 012702 000101               MOV    #65.,R2           ;SET OR AND RESET IR
010610 012737 000000 001434        MOV    #0,RMDBO          ;WRITE ZEROS IN DATA SILO
010616 112737 000022 001545        MOVB   #RMDB,PUTINX      ;SETUP REGISTER INDEX
010624 112737 000200 001546        MOVB   #200,PUTINX+1
010632                      2$:
010632 004737 043016               JSR    PC,PUT            ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
010636 000404                      BR     3$                ;GO TO 3$ IF NO ERROR
010640 000240                      NOP                      ;RETURN HERE IF ERROR
010642 104000                      EMT                      ;ERROR # DEFINED BY PUT SUBROUTINE
010644 000137 011402               JMP    15$               ;GO TO 15$ IF ERROR
010650 005302               3$:    DEC    R2                ;DECREMENT COUNT
010652 001367                      BNE    2$                ;WRITE SILO AGAIN
010654 012737 177777 001416        MOV    #-1,RMBAO         ;RMBA = ALL 1'S
010662 012737 177777 001426        MOV    #-1,RMER1O        ;RMER1 = ALL 1'S
010670 012737 177777 001454        MOV    #-1,RMER2O        ;RMER2 = ALL 1'S
```

```
                                                MOV     #DMD!DBEN,RMMR10        ;SET DIAGNOSTIC MODE
                                                MOV     #003577,RMCS10
                                                MOV     #021037,RMCS20

                                                MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
                                                MOVB    #RMBA,(R2)+
                                                MOVB    #RMER1,(R2)+
                                                MOVB    #RMER2,(R2)+
                                                MOVB    #RMMR1,(R2)+
                                                MOVB    #RMCS1,(R2)+
                                                MOVB    #RMCS2,(R2)+
                                                MOVB    #200,(R2)+
                                                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                                BR      4$              ;GO TO 4$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                                JMP     15$             ;GO TO 15$ IF ERROR
                                4$:             RESET                   ;UNIBUS INITIALIZE
                                                JSR     PC,$TKINT       ;INITIALIZE CONSOLE
                                                MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
                                                JSR     PC,GETSTS       ;GO SET UP FOR STATUS FETCH

                                                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                                BR      5$              ;GO TO 5$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                                                BR      15$             ;SKIP REMAINDER OF TEST

                                5$:             MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
                                                BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
                                                MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
                                                CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
                                                BEQ     6$              ;BRANCH IF EQUAL
                                                EMT     115

                                6$:             CLR     $GDDAT          ;VERIFY RMBA IS ZERO
                                                MOV     RMBAI,$BDDAT
                                                BEQ     7$              ;BRANCH IF ZERO
                                                EMT     116

                                7$:             MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
                                                CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
                                                MOVB    (R1),(SP)
                                                BIS     #IR,(SP)
                                                MOV     (SP)+,$GDDAT
                                                CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
                                                BEQ     8$              ;BRANCH IF EQUAL
                                                EMT     117

                                8$:             CLR     $GDDAT          ;VERIFY RMER1
                                                MOV     RMER1I,$BDDAT
                                                BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
                                                BEQ     9$              ;BRANCH IF ZERO
                                                EMT     120

                                9$:             MOV     RMASI,$BDDAT    ;VERIFY RMAS
                                                CLR     R2              ;CLEAR ALL BUT THIS
```

```
    170 011172  116102  000001                    MOVB    1(R1),R2        ;DRIVES ATTENTION BIT
    171 011176  000302                            SWAB    R2
    172 011200  005102                            COM     R2
    173 011202  000240                            NOP
    174 011204  040237  001142                    BIC     R2,$BDDAT
    175 011210  001401                            BEQ     10$             ;BRANCH IF ITS 0
    176 011212  104121                            EMT     121
    177
    178 011214  013737  001362  001142  10$:     MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
    179 011222  042737  000046  001142            BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
    180 011230  012737  000010  001140            MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
    181 011236  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
    182 011244  001401                            BEQ     11$             ;BRANCH IF 0
    183 011246  104122                            EMT     122
    184
    185 011250  005037  001140            11$:    CLR     $GDDAT          ;EXPECT ZEROS
    186 011254  013737  001404  001142            MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2 = 0
    187 011262  001401                            BEQ     12$
    188 011264  104124                            EMT     124
    189
    190 011266  013737  001376  001142  12$:     MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
    191 011274  042737  140000  001142            BIC     #RQA!RQB,$BDDAT
    192 011302  012737  011777  001140            MOV     #TST!1777,$GDDAT        ;EXPECT TEST,TAG BIT ON
    193 011310  023737  001140  001142            CMP     $GDDAT,$BDDAT
    194 011316  001401                            BEQ     13$
    195 011320  104125                            EMT     125
    196
    197 011322  005037  001140            13$:    CLR     $GDDAT          ;EXPECT ALL ZEROS
    198 011326  013737  001400  001142            MOV     RMER2I,$BDDAT   ;VERIFY RMER2
    199 011334  042737  040200  001142            BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
    200 011342  001401                            BEQ     14$             ;BRANCH IF OTHER BITS 0
    201 011344  104173                            EMT     173
    202 011346  013737  001350  001142  14$:     MOV     RMDSI,$BDDAT    ;CHECK DRIVE STATUS REGISTER
    203 011354  042737  177177  001142            BIC     #^C<DPR!DRY>,$BDDAT
    204 011362  012737  000600  001140            MOV     #DPR!DRY,$GDDAT ;EXPECTED STATUS
    205 011370  023737  001142  001140            CMP     $BDDAT,$GDDAT   ;COMPARE EXPECTED & RECEIVED STATUS
    206 011376  001401                            BEQ     15$             ;DRIVE STATUS IS OK
    207 011400  104123                            EMT     123
    208 011402                          15$:
    209
    210                                 ;;*************************************************************************
                                        ;*TEST 4         CONTROLLER CLEAR TEST
                                        ;;*************************************************************************
        011402                          TST4:
        011402  000004                          SCOPE                   ;SCOPE CALL
        011404  000240                          NOP                     ;START OF TEST
        011406  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
        011412  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        011416  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
        011422  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    211
    212 011430  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
        011434  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
        011436  000240                          NOP                     ;RETURN HERE IF ERROR
        011440  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
        011442  000137  011740                  JMP     8$              ;GO TO 8$ IF ERROR
    213 011446                          1$:
```

```
                                                MOV     #65.,R2
                                                MOV     #0,RMDB0        ;WRITE ZEROS IN DATA SILO
                                                MOVB    #RMDB,PUTINX    ;SETUP REGISTER INDEX TABLE
                                                MOVB    #200,PUTINX+1
                                        2$:
                                                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                                BR      3$              ;GO TO 3$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                                JMP     8$              ;GO TO 8$ IF ERROR
                                        3$:     DEC     R2              ;DECREMENT COUNT
                                                BNE     2$              ;AND WRITE SILO AGAIN IF NOT DONE
                                                MOV     #-1,RMBA0
                                                MOV     #-1,RMER10
                                                MOV     #-1,RMER20
                                                MOV     #DMD!DBEN,RMMR10
                                                MOV     #003577,RMCS10
                                                MOV     #021037,RMCS20
                                                MOV     #PUTINX,R2      ;R2 = ADDRESS OF INDEX TABLE
                                                MOVB    #RMBA,(R2)+
                                                MOVB    #RMER1,(R2)+
                                                MOVB    #RMER2,(R2)+
                                                MOVB    #RMMR1,(R2)+
                                                MOVB    #RMCS1,(R2)+
                                                MOVB    #RMCS2,(R2)+
                                                MOVB    #200,(R2)+
                                                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                                BR      4$              ;GO TO 4$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                                BR      7$

                                        4$:
                                                JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                                                BR      5$              ;GO TO 5$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                                                JMP     8$              ;GO TO 8$ IF ERROR
                                        5$:     JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS

                                                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                                BR      6$              ;GO TO 6$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                                                JMP     8$              ;GO TO 8$ IF ERROR
                                        6$:
                                                JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
                                                BR      7$              ;GO TO 7$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY CLRSTS SUBROUTINE
                                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                                JMP     8$              ;GO TO 8$ IF ERROR
                                        7$:
                                                MOV     #NOP,RMCS10     ;CHANGE  FUNCTION CODE FOR ERROR CHECK

                                                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                                BR      8$              ;GO TO 8$ IF NO ERROR
```

```
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
250                             8$:
251
252                             ;;**************************************************************************
                                ;*TEST 5         ERROR CLEAR TEST
                                ;;**************************************************************************
                                TST5:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
253
254                                     JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                                        BR      1$              ;GO TO 1$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                                        JMP     5$              ;GO TO 5$ IF ERROR
255                             1$:
256                                     MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                                        MOV     #TRE,RMCS10     ;SET RMCS1 OUTPUT BUFFER = TRE
                                        JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                                        BR      2$              ;GO TO 2$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                                        JMP     5$              ;GO TO 5$ IF ERROR
257                             2$:
258                                     MOVB    #RMCS1,GETINX
259                                     MOVB    #RMCS2,GETINX+1
260                                     MOVB    #200,GETINX+2
261
262                                     JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                        BR      3$              ;GO TO 3$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
263                                     BR      5$              ;SKIP REST OF TEST
264                             3$:     MOV     RMCS11,$BDDAT    ;CHECK TRE & MCPE
265                                     CLR     $GDDAT          ;EXPECT 0'S
266                                     BIC     #^C<TRE!MCPE>,$BDDAT
267                                     BEQ     4$              ;BRANCH IF TRE & MCPE = 0
268                                     EMT     137
269
270                             4$:     MOV     RMCS21,$BDDAT   ;CHECK RMCS2
271                                     BIC     #^C<WCE!UPE!NED!PGE!MXF!MDPE>,$BDDAT
272                                     BEQ     5$
273                                     EMT     140
274
275                             5$:
276
277                             ;;**************************************************************************
                                ;*TEST 6         DRIVE STATUS TEST
                                ;;**************************************************************************
                                TST6:
```

```
                012146  000004                          SCOPE                   ;SCOPE CALL
                012150  000240                          NOP                     ;START OF TEST
                012152  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                012156  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                012162  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                012166  012737  000006  001226          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    278
    279         012174  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                012200  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                012202  000240                          NOP                     ;RETURN HERE IF ERROR
                012204  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                012206  000137  012462                  JMP     8$              ;GO TO 8$ IF ERROR
    280         012212                          1$:
    281         012212  004737  042462                  JSR     PC,GETSTS
    282
    283         012216  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                012222  000403                          BR      2$              ;GO TO 2$ IF NO ERROR
                012224  000240                          NOP                     ;RETURN HERE IF ERROR
                012226  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
    284         012230  000514                          BR      8$              ;SKIP REST OF TEST
    285         012232  032737  010000  001350  2$:     BIT     #MOL,RMDSI      ;MEDIUM ON LINE??
    286         012240  001016                          BNE     3$              ;YES!!
    287         012242  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
    288         012250  042737  162000  001140          BIC     #ATA!ERR!PIP!LBT,$GDDAT
    289         012256  052737  010000  001140          BIS     #MOL,$GDDAT
    290         012264  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
    291         012272  104151                          EMT     151             ;MOL STATUS INCORRECT
    292         012274  000440                          BR      5$
    293
    294         012276  032737  000200  001400  3$:     BIT     #DVC,RMER2I     ;IS THERE A DEVICE CHECK??
    295         012304  001422                          BEQ     4$              ;BR IF NO
    296         012306  013737  001400  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
    297         012314  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
    298         012320  104152                          EMT     152             ;DRIVE FAULT
    299
    300         012322  032737  040000  001352          BIT     #UNS,RMER1I     ;IS UNS SET??
    301         012330  001022                          BNE     5$              ;YES!!
    302         012332  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
    303         012340  012737  040000  001140          MOV     #UNS,$GDDAT     ;EXPECTED STATUS
    304         012346  104153                          EMT     153             ;"DVC" IS SET BUT "UNS" IS NOT
    305         012350  000412                          BR      5$
    306
    307         012352  032737  040000  001352  4$:     BIT     #UNS,RMER1I     ;IS UNS SET??
    308         012360  001410                          BEQ     6$              ;NO!!
    309         012362  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
    310         012370  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
    311         012374  104154                          EMT     154             ;"DVC" IS NOT SET, BUT "UNS" IS
    312         012376  000137  041256          5$:     JMP     $EOSP           ;SKIP REST OF TEST
    313
    314         012402  032737  004000  001350  6$:     BIT     #WRL,RMDSI      ;IS WRITE PROTECT ON??
    315         012410  001412                          BEQ     7$              ;NO!!
    316         012412  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
    317         012420  042737  166076  001140          BIC     #^C<MOL!PGM!DPR!DRY!VV!OM>,$GDDAT
    318         012426  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
    319         012434  104175                          EMT     175             ;SELECTED DEVICE IN WRITE PROTECT
    320
    321         012436  032737  040000  001400  7$:     BIT     #SKI,RMER2I     ;IS SKI SET??
```

```
                                        BEQ     8$                      ;NO!!
                                        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
                                        CLR     $GDDAT                  ;EXPECTED STATUS
                                        EMT     205                     ;PERSISTENT SEEK INCOMPLETE ERROR
                                8$:

                                ;;*******************************************************************
                                ;*TEST 7         PRIMARY/SECONDARY ERROR TEST
                                ;;*******************************************************************
                                TST7:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,CNTCLR               ;GO ISSUE CONTROLLER CLEAR
                                        BR      1$                      ;GO TO 1$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        EMT                             ;ERROR NUMBER DEFINED BY SUBROUTINE
                                        JMP     5$                      ;GO TO 5$ IF ERROR
                                1$:
                                        MOVB    #RMCS1,PUTINX           ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1           ;SET TERMINATOR BYTE
                                        MOV     #PAKACK!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
                                        JSR     PC,PUT                  ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                                        BR      2$                      ;GO TO 2$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        EMT                             ;ERROR DEFINED BY PUT SUBROUTINE
                                        BR      5$                      ;SKIP REST OF TEST IF ERROR
                                2$:     JSR     PC,GETSTS               ;SETUP FOR STATUS FETCH
                                        JSR     PC,TIMOUT               ;WAIT FOR COMPLETION OF NOP

                                        JSR     PC,GET                  ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                        BR      3$                      ;GO TO 3$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        EMT                             ;ERROR # DEFINED BY GET SUBROUTINE
                                        BR      5$                      ;SKIP REST OF TEST IF ERROR
                                3$:
                                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                                        BR      4$                      ;GO TO 4$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        EMT                             ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        BR      5$                      ;SKIP REST OF TEST IF ERROR
                                4$:
                                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
                                        BR      5$                      ;GO TO 5$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        EMT                             ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                5$:

                                ;;*******************************************************************
                                ;*TEST 10        DIAGNOSTIC MODE TEST
                                ;;*******************************************************************
```

```
                012642                          TST10:
                012642  000004                          SCOPE                   ;SCOPE CALL
                012644  000240                          NOP                     ;START OF TEST
                012646  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                012652  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                012656  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                012662  012737  000010  001226          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
            347
            348 012670  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                012674  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                012676  000240                          NOP                     ;RETURN HERE IF ERROR
                012700  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                012702  000137  013714                  JMP     21$             ;GO TO 21$ IF ERROR
            349 012706                          1$:
            350 012706  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                012714  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                012722  012737  000001  001436          MOV     #DMD,RMMR1O     ;SET RMMR1 OUTPUT BUFFER = DMD
                012730  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                012734  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                012736  000240                          NOP                     ;RETURN HERE IF ERROR
                012740  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                012742  000137  013714                  JMP     21$             ;GO TO 21$ IF ERROR
            351 012746  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
            352
            353 012752  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                012756  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                012760  000240                          NOP                     ;RETURN HERE IF ERROR
                012762  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                012764  000137  013714                  JMP     21$             ;GO TO 21$ IF ERROR
            354 012770  032737  000001  001362  3$:     BIT     #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
            355 012776  001011                          BNE     4$              ;YES!!
            356 013000  012737  000001  001140          MOV     #DMD,$GDDAT     ;EXPECTED STATUS
            357 013006  013737  001362  001142          MOV     RMMR1I,$BDDAT   ;RECEIVED STATUS
            358 013014  104176                          EMT     176
            359 013016  000137  013714                  JMP     21$             ;SKIP REST OF TEST IF DMD = 0
            360 013022  032737  010000  001350  4$:     BIT     #MOL,RMDSI      ;IS 'MOL" = 0   ??
            361 013030  001411                          BEQ     5$              ;YES!!
            362 013032  013737  001350  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
            363 013040  042737  167777  001142          BIC     #^CMOL,$BDDAT
            364 013046  005037  001140                  CLR     $GDDAT          ;SETUP GOOD DATA FOR TYPEOUT
            365 013052  104170                          EMT     170
            366 013054  032737  020000  001350  5$:     BIT     #PIP,RMDSI      ;IS PIP  SET??
            367 013062  001012                          BNE     6$              ;YES!!
            368 013064  013737  001350  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
            369 013072  042737  157777  001142          BIC     #^CPIP,$BDDAT
            370 013100  012737  020000  001140          MOV     #PIP,$GDDAT     ;EXPECTED PIP SET
            371 013106  104200                          EMT     200
            372 013110  032737  004000  001350  6$:     BIT     #WRL,RMDSI      ;IS WRITE LOCK OFF??
            373 013116  001411                          BEQ     7$              ;YES!!
            374 013120  013737  001350  001142          MOV     RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
            375 013126  042737  173777  001142          BIC     #^CWRL,$BDDAT
            376 013134  005037  001140                  CLR     $GDDAT          ;EXPECTED WRL = 0
            377 013140  104201                          EMT     201
            378 013142  032737  040000  001400  7$:     BIT     #SKI,RMER2I     ;IS SKI = 0
            379 013150  001411                          BEQ     8$              ;YES!!
            380 013152  013737  001400  001142          MOV     RMER2I,$BDDAT   ;SETUP BAD DATA FOR TYPEOUT
            381 013160  042737  137777  001142          BIC     #^CSKI,$BDDAT
```

```
382 013166 005037 001140           CLR   $GDDAT          ;SKI SHOULD BE 0
383 013172 104202                  EMT   202
384 013174 032737 000200 001400 8$: BIT   #DVC,RMER2I     ;IS DEVICE CHECK = 0??
385 013202 001411                  BEQ   9$              ;YES!!
386 013204 013737 001400 001142    MOV   RMER2I,$BDDAT   ;SETUP BAD DATA FOR  TYPEOUT
387 013212 042737 177577 001142    BIC   #^CDVC,$BDDAT
388 013220 005037 001140           CLR   $GDDAT          ;DVC SHOULD BE 0
389 013224 104203                  EMT   203
390 013226                      9$:
391 013226 112737 000024 001545    MOVB  #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
    013234 112737 000200 001546    MOVB  #200,PUTINX+1   ;SET TERMINATOR BYTE
    013242 012737 001711 001436    MOV   #DMD!MUR!MOC!MSER!MDF!MWP,RMMR1O        ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!M
    013250 004737 043016           JSR   PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
    013254 000404                  BR    10$             ;GO TO 10$ IF NO ERROR
    013256 000240                  NOP                   ;RETURN HERE IF ERROR
    013260 104000                  EMT                   ;ERROR DEFINED BY PUT SUBROUTINE
    013262 000137 013714           JMP   21$             ;GO TO 21$ IF ERROR
392 013266                     10$:
393 013266 004737 042546           JSR   PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
    013272 000404                  BR    11$             ;GO TO 11$ IF NO ERROR
    013274 000240                  NOP                   ;RETURN HERE IF ERROR
    013276 104000                  EMT                   ;ERROR # DEFINED BY GET SUBROUTINE
    013300 000137 013714           JMP   21$             ;GO TO 21$ IF ERROR
394 013304 032737 000001 001362 11$: BIT  #DMD,RMMR1I     ;IS  DIAGNOSTIC MODE SET??
395 013312 001011                  BNE   12$             ;YES!!
396 013314 012737 000001 001140    MOV   #DMD,$GDDAT     ;EXPECTED STATUS
397 013322 013737 001362 001142    MOV   RMMR1I,$BDDAT   ;RECEIVED STATUS
398 013330 104176                  EMT   176
399 013332 000137 013714           JMP   21$             ;SKIP REST OF TEST IF DMD = 0
400 013336 032737 010000 001350 12$: BIT  #MOL,RMDSI     ;IS MOL = 1??
401 013344 001012                  BNE   13$             ;YES!!
402 013346 013737 001350 001142    MOV   RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
403 013354 042737 167777 001142    BIC   #^CMOL,$BDDAT
404 013362 012737 010000 001140    MOV   #MOL,$GDDAT     ;EXPECTED MOL = 1
405 013370 104170                  EMT   170
406 013372 032737 020000 001350 13$: BIT  #PIP,RMDSI     ;IS PIP = 0 ?
407 013400 001411                  BEQ   14$             ;YES!!
408 013402 013737 001350 001142    MOV   RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
409 013410 042737 157777 001142    BIC   #^CPIP,$BDDAT
410 013416 005037 001140           CLR   $GDDAT          ;EXPECTED PIP STATUS
411 013422 104200                  EMT   200
412 013424 032737 004000 001350 14$: BIT  #WRL,RMDSI     ;IS WRL SET??
413 013432 001012                  BNE   15$             ;YES!!
414 013434 013737 001350 001142    MOV   RMDSI,$BDDAT    ;SETUP BAD DATA FOR TYPEOUT
415 013442 042737 173777 001142    BIC   #^CWRL,$BDDAT
416 013450 012737 004000 001140    MOV   #WRL,$GDDAT     ;EXPECTED GOOD STATUS
417 013456 104201                  EMT   201
418 013460 032737 040000 001400 15$: BIT  #SKI,RMER2I     ;IS SKI SET??
419 013466 001012                  BNE   16$             ;YES!!
420 013470 013737 001400 001142    MOV   RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
421 013476 042737 137777 001142    BIC   #^CSKI,$BDDAT
422 013504 012737 040000 001140    MOV   #SKI,$GDDAT     ;EXPECTED SKI  ON
423 013512 104202                  EMT   202
424 013514 032737 000200 001400 16$: BIT  #DVC,RMER2I     ;IS DVC SET  ??
425 013522 001012                  BNE   17$             ;YES!!
426 013524 013737 001400 001142    MOV   RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
427 013532 042737 177577 001142    BIC   #^CDVC,$BDDAT
```

```
    428 013540  012737  000200  001140          MOV     #DVC,$GDDAT     ;EXPECTED DVC ON
    429 013546  104203                          EMT     203
    430 013550  032737  000100  001350  17$:    BIT     #VV,RMDSI       ;MUR SHOULD HAVE RESET VOLUME VALID
    431 013556  001411                          BEQ     18$             ;BRANCH IF IT DID
    432 013560  013737  001350  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
    433 013566  042737  177677  001142          BIC     #^CVV,$BDDAT
    434 013574  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    435 013600  104204                          EMT     204
    436 013602                          18$:
    437 013602  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
        013610  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
        013616  012737  000011  001412          MOV     #DRVCLR!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = DRVCLR!GO
        013624  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
        013630  000404                          BR      19$             ;GO TO 19$ IF NO ERROR
        013632  000240                          NOP                     ;RETURN HERE IF ERROR
        013634  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
        013636  000137  013714                  JMP     21$             ;GO TO 21$ IF ERROR
    438 013642                          19$:
    439 013642  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
        013646  000404                          BR      20$             ;GO TO 20$ IF NO ERROR
        013650  000240                          NOP                     ;RETURN HERE IF ERROR
        013652  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
        013654  000137  013714                  JMP     21$             ;GO TO 21$ IF ERROR
    440 013660  032737  002000  001400  20$:    BIT #LBC,RMER2I         ;IS LBC SET ??
    441 013666  001012                          BNE 21$                 ;YES!!
    442 013670  013737  001400  001142          MOV RMER2I,$BDDAT       ;BAD DATA FOR TYPEOUT
    443 013676  012737  002000  001140          MOV #LBC, $GDDAT        ;GOOD DATA FOR TYPEOUT
    444 013704  042737  002000  001142          BIC #LBC, $BDDAT
    445 013712  104262                          EMT     262
    446 013714                          21$:
    447
    448                                 ;;*************************************************************************
                                        ;*TEST 11        PACK ACKNOWLEDGE TEST
                                        ;;*************************************************************************
        013714                          TST11:
        013714  000004                          SCOPE                   ;SCOPE CALL
        013716  000240                          NOP                     ;START OF TEST
        013720  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
        013724  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        013730  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
        013734  012737  000011  001226          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    449
    450 013742  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
        013746  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
        013750  000240                          NOP                     ;RETURN HERE IF ERROR
        013752  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
        013754  000137  014214                  JMP     8$              ;GO TO 8$ IF ERROR
    451 013760                          1$:
    452 013760  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
        013766  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
        013774  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
        014002  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
        014006  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
        014010  000240                          NOP                     ;RETURN HERE IF ERROR
        014012  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
        014014  000137  014214                  JMP     8$              ;GO TO 8$ IF ERROR
    453 014020                          2$:
```

```
                                        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                                        MOV     #0,RMMR1O       ;SET RMMR1 OUTPUT BUFFER = 0
                                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                                        BR      3$              ;GO TO 3$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                                        JMP     8$              ;GO TO 8$ IF ERROR
                                3$:
                                        MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                                        MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                                        MOV     #PAKACK!GO,RMCS1O       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
                                        JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                                        BR      4$              ;GO TO 4$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                                        JMP     8$              ;GO TO 8$ IF ERROR
                                4$:     JSR     PC,GETSTS
                                        JSR     PC,TIMOUT       ;WAIT FOR COMPLETION

                                        JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                        BR      5$              ;GO TO 5$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                                        BR      8$              ;SKIP REMAINDER OF TEST
                                5$:
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      6$              ;GO TO 6$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        BR      8$              ;SKIP TO NEXT TEST
                                6$:
                                        JSR     PC,ACKSTS       ;GO VERIFY PACK ACKNOWLEDGE
                                        BR      7$              ;GO TO 7$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY ACKSTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        BR      8$              ;SKIP TO NEXT TEST

                                7$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      8$              ;GO TO 8$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                8$:

                                ;;*******************************************************************
                                ;*TEST 12        RECALIBRATE TEST
                                ;;*******************************************************************
                                TST12:
                                        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
```

```
T12     RECALIBRATE TEST

        014234  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   474
   475  014242  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
        014246  050020                          .WORD   050020          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
        014250  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
        014252  000240                          NOP                     ;RETURN HERE IF ERROR
        014254  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
        014256  000137  014424                  JMP     6$              ;GO TO 6$ IF ERROR
   476  014262                          1$:
   477  014262  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
        014270  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
        014276  012737  000007  001412          MOV     #RECAL!GO,RMCS1O        ;SET RMCS1 OUTPUT BUFFER = RECAL!GO
        014304  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
        014310  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
        014312  000240                          NOP                     ;RETURN HERE IF ERROR
        014314  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
        014316  000137  014424                  JMP     6$              ;GO TO 6$ IF ERROR
   478  014322  004737  042462          2$:     JSR     PC,GETSTS       ;GO SETUP FOR STATUS FETCH
   479  014326  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
   480
   481  014332  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
        014336  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
        014340  000240                          NOP                     ;RETURN HERE IF ERROR
        014342  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
        014344  000137  014424                  JMP     6$              ;GO TO 6$ IF ERROR
   482  014350                          3$:
   483  014350  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
        014354  000405                          BR      4$              ;GO TO 4$ IF NO ERROR
        014356  000240                          NOP                     ;RETURN HERE IF ERROR
        014360  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
        014362  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
        014364  000137  014424                  JMP     6$              ;GO TO 6$ IF ERROR
   484  014370                          4$:
   485  014370  004737  054260                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
        014374  000405                          BR      5$              ;GO TO 5$ IF NO ERROR
        014376  000240                          NOP                     ;RETURN HERE IF ERROR
        014400  104000                          EMT                     ;ERROR # DEFINED BY RCLSTS SUBROUTINE
        014402  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
        014404  000137  014424                  JMP     6$              ;GO TO 6$ IF ERROR
   486  014410                          5$:
   487  014410  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
        014414  000403                          BR      6$              ;GO TO 6$ IF NO ERROR
        014416  000240                          NOP                     ;RETURN HERE IF ERROR
        014420  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
        014422  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
   488  014424                          6$:
   489
   490                                  ;;*************************************************************************
                                        ;*TEST 13        ABORT RECALIBRATE TEST
                                        ;;*************************************************************************
        014424                          TST13:
        014424  000004                          SCOPE                   ;SCOPE CALL
        014426  000240                          NOP                     ;START OF TEST
        014430  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

T13 ABORT RECALIBRATE TEST

```
                014434  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                014440  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                014444  012737  000001  001206          MOV     #1,$TIMES       ;;DO 1 ITERATION
                014452  012737  000013  001226          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
            491
            492 014460  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                014464  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                014466  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                014470  000240                          NOP                     ;RETURN HERE IF ERROR
                014472  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                014474  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
            493 014500                          1$:
            494 014500  112737  000014  001545          MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
                014506  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                014514  012737  040000  001426          MOV     #UNS,RMER10     ;SET RMER1 OUTPUT BUFFER = UNS
                014522  004737  043016                  JSR     PC,PUT          ;GO WRITE RMER1 VIA PUT SUBROUTINE
                014526  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                014530  000240                          NOP                     ;RETURN HERE IF ERROR
                014532  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                014534  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
            495 014540                          2$:
            496 014540  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                014546  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                014554  012737  000007  001412          MOV     #RECAL!GO,RMCS10        ;SET RMCS1 OUTPUT BUFFER = RECAL!GO
                014562  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                014566  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                014570  000240                          NOP                     ;RETURN HERE IF ERROR
                014572  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                014574  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
            497 014600  004737  042462          3$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
            498
            499 014604  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                014610  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                014612  000240                          NOP                     ;RETURN HERE IF ERROR
                014614  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                014616  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
            500 014622  032737  020000  001350  4$:     BIT     #PIP,RMDSI      ;DID THE DRIVE RECALIBRATE??
            501 014630  001401                          BEQ     5$              ;NO!!
            502 014632  104214                          EMT     214
            503 014634  004737  043370          5$:     JSR     PC,TIMOUT       ;WAIT FOR  COMPLETION
            504
            505 014640  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                014644  000404                          BR      6$              ;GO TO 6$ IF NO ERROR
                014646  000240                          NOP                     ;RETURN HERE IF ERROR
                014650  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                014652  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
            506 014656                          6$:
            507 014656  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                014662  000405                          BR      7$              ;GO TO 7$ IF NO ERROR
                014664  000240                          NOP                     ;RETURN HERE IF ERROR
                014666  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
```

```
                014670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                014672  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
    508         014676                          7$:
    509         014676  004737  060210                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
                014702  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
                014704  000240                          NOP                     ;RETURN HERE IF ERROR
                014706  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
                014710  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                014712  000137  014732                  JMP     9$              ;GO TO 9$ IF ERROR
    510         014716                          8$:
    511         014716  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                014722  000403                          BR      9$              ;GO TO 9$ IF NO ERROR
                014724  000240                          NOP                     ;RETURN HERE IF ERROR
                014726  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                014730  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    512         014732                          9$:
    513
    514                                         ;;*****************************************************************
                                                ;*TEST 14        IVC RECALIBRATE TEST
                                                ;;*****************************************************************
                014732                          TST14:
                014732  000004                          SCOPE                   ;SCOPE CALL
                014734  000240                          NOP                     ;START OF TEST
                014736  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                014742  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                014746  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                014752  012737  000001  001206          MOV     #1,$TIMES       ;;DO 1 ITERATION
                014760  012737  000014  001226          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    515
    516         014766  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                014772  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                014774  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                014776  000240                          NOP                     ;RETURN HERE IF ERROR
                015000  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                015002  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
    517         015006                          1$:
    518         015006  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                015014  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                015022  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
                015030  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                015034  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                015036  000240                          NOP                     ;RETURN HERE IF ERROR
                015040  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                015042  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
    519         015046                          2$:
    520         015046  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                015054  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                015062  012737  000000  001436          MOV     #0,RMMR10       ;SET RMMR1 OUTPUT BUFFER = 0
                015070  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                015074  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                015076  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
                015100  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                015102  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
521             015106                          3$:
522             015106  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                015114  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                015122  012737  000007  001412          MOV     #RECAL!GO,RMCS10        ;SET RMCS1 OUTPUT BUFFER = RECAL!GO
                015130  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                015134  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                015136  000240                          NOP                     ;RETURN HERE IF ERROR
                015140  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                015142  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
523             015146  004737  042462          4$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
524             015152  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
525
526             015156  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                015162  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                015164  000240                          NOP                     ;RETURN HERE IF ERROR
                015166  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                015170  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
527             015174                          5$:
528             015174  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                015200  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
                015202  000240                          NOP                     ;RETURN HERE IF ERROR
                015204  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                015206  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                015210  000137  015242                  JMP     8$              ;GO TO 8$ IF ERROR
529             015214  032737  010000  001400  6$:     BIT     #IVC,RMER2I     ;IVC SHOULD BE SET
530             015222  001001                          BNE     7$              ;IT IS - GO CHECK SECONDARY ERRORS
531             015224  104215                          EMT     215
532             015226                          7$:
533             015226  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                015232  000403                          BR      8$              ;GO TO 8$ IF NO ERROR
                015234  000240                          NOP                     ;RETURN HERE IF ERROR
                015236  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                015240  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
534             015242                          8$:
535
536                                             ;;*****************************************************************
                                                ;*TEST 15        IAE RECALIBRATE TEST
                                                ;;*****************************************************************
                015242                          TST15:
                015242  000004                          SCOPE                   ;SCOPE CALL
                015244  000240                          NOP                     ;START OF TEST
                015246  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                015252  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                015256  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                015262  012737  000001  001206          MOV     #1,$TIMES       ;;DO 1 ITERATION
                015270  012737  000015  001226          MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
537
538             015276  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                015302  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
```

```
                015304  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                015306  000240                          NOP                     ;RETURN HERE IF ERROR
                015310  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                015312  000137  015504                  JMP     6$              ;GO TO 6$ IF ERROR
            539 015316                          1$:
            540 015316  012737  177777  001446          MOV     #-1,RMDCO       ;RMDC WILL BE ALL ONES
            541 015324  012737  177777  001420          MOV     #-1,RMDAO       ;RMDA WILL BE ALL ONES
            542 015332  012737  000007  001412          MOV     #RECAL!GO,RMCS10
            543 015340  012702  001545                  MOV     #PUTINX,R2      ;R2 POINTS TO PUT INDEX TABLE
            544 015344  112722  000034                  MOVB    #RMDC,(R2)+     ;SETUP PUT INDEX TABLE
            545 015350  112722  000006                  MOVB    #RMDA,(R2)+
            546 015354  112722  000000                  MOVB    #RMCS1,(R2)+
            547 015360  112722  000200                  MOVB    #200,(R2)+
            548 015364  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                015370  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                015372  000240                          NOP                     ;RETURN HERE IF ERROR
                015374  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                015376  000137  015504                  JMP     6$              ;GO TO 6$ IF ERROR
            549 015402  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
            550 015406  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
            551
            552 015412  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                015416  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                015420  000240                          NOP                     ;RETURN HERE IF ERROR
                015422  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                015424  000137  015504                  JMP     6$              ;GO TO 6$ IF ERROR
            553 015430                          3$:
            554 015430  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                015434  000405                          BR      4$              ;GO TO 4$ IF NO ERROR
                015436  000240                          NOP                     ;RETURN HERE IF ERROR
                015440  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                015442  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                015444  000137  015504                  JMP     6$              ;GO TO 6$ IF ERROR
            555 015450                          4$:
            556 015450  004737  054260                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
                015454  000405                          BR      5$              ;GO TO 5$ IF NO ERROR
                015456  000240                          NOP                     ;RETURN HERE IF ERROR
                015460  104000                          EMT                     ;ERROR # DEFINED BY RCLSTS SUBROUTINE
                015462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                015464  000137  015504                  JMP     6$              ;GO TO 6$ IF ERROR
            557 015470                          5$:
            558 015470  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                015474  000403                          BR      6$              ;GO TO 6$ IF NO ERROR
                015476  000240                          NOP                     ;RETURN HERE IF ERROR
                015500  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                015502  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
            559 015504                          6$:
            560
            561                                 ;;*****************************************************************
                                                ;*TEST 16        RECALIBRATE AT OFFSET
                                                ;;*****************************************************************
                015504                          TST16:
                015504  000004                          SCOPE                   ;SCOPE CALL
                015506  000240                          NOP                     ;START OF TEST
                015510  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                015514  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                015520  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
                015524  012737  000001  001206          MOV     #1,$TIMES       ;;DO 1 ITERATION
                015532  012737  000016  001226          MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
     562
     563        015540  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                015544  050020                          .WORD   050020          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                015546  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                015550  000240                          NOP                     ;RETURN HERE IF ERROR
                015552  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                015554  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
     564        015560                          1$:
     565        015560  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                015566  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                015574  012737  000015  001412          MOV     #OFFSET!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
                015602  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                015606  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                015610  000240                          NOP                     ;RETURN HERE IF ERROR
                015612  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                015614  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
     566        015620                          2$:
     567        015620  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                015626  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                015634  012737  000007  001412          MOV     #RECAL!GO,RMCS10        ;SET RMCS1 OUTPUT BUFFER = RECAL!GO
                015642  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                015646  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                015650  000240                          NOP                     ;RETURN HERE IF ERROR
                015652  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                015654  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
     568        015660                          3$:
     569
     570                                        ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
                015660  004737  042462                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
     571
     572                                        ;WAIT FOR COMMAND TO COMPLETE
                015664  004737  043370                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
     573
     574        015670  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                015674  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                015676  000240                          NOP                     ;RETURN HERE IF ERROR
                015700  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                015702  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
     575        015706                          4$:
     576        015706  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                015712  000405                          BR      5$              ;GO TO 5$ IF NO ERROR
                015714  000240                          NOP                     ;RETURN HERE IF ERROR
                015716  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                015720  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                015722  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
     577        015726                          5$:
     578        015726  004737  054260                  JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE OPERATION
                015732  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
                015734  000240                          NOP                     ;RETURN HERE IF ERROR
                015736  104000                          EMT                     ;ERROR # DEFINED BY RCLSTS SUBROUTINE
                015740  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                015742  000137  015762                  JMP     7$              ;GO TO 7$ IF ERROR
```

```
                                6$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      7$              ;GO TO 7$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                7$:

                                ;;*******************************************************************
                                ;*TEST 17        DRIVE CLEAR TEST
                                ;;*******************************************************************
                                TST17:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
                                        BR      1$              ;GO TO 1$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     6$              ;GO TO 6$ IF ERROR
                                1$:
                                        MOV     #0,RMDA0
                                        MOV     #DRVCLR!GO,RMCS10
                                        MOV     #-1,RMER10
                                        MOV     #-1,RMER20
                                        BIC     #LSC,RMER20     ;DELETE FOR PASS 2 ETCH
                                        MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
                                        MOVB    #RMDA,(R2)+     ;RMDA CLEARS LBT
                                        MOVB    #RMER2,(R2)+
                                        MOVB    #RMER1,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+
                                        JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                        BR      2$              ;GO TO 2$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                        BR      6$              ;ESCAPE IF ERROR
                                2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
                                        JSR     PC,TIMOUT       ;WAIT FOR DRIVE CLEAR TO COMPLETE

                                        JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                        BR      3$              ;GO TO 3$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                                        BR      6$              ;SKIP REMAINDER OF TEST
                                3$:
```

```
606 016156  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
    016162  000405                  BR      4$              ;GO TO 4$ IF NO ERROR
    016164  000240                  NOP                     ;RETURN HERE IF ERROR
    016166  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
    016170  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    016172  000137  016232          JMP     6$              ;GO TO 6$ IF ERROR
607 016176                  4$:
608 016176  004737  056022          JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR
    016202  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
    016204  000240                  NOP                     ;RETURN HERE IF ERROR
    016206  104000                  EMT                     ;ERROR # DEFINED BY DRVSTS SUBROUTINE
    016210  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    016212  000137  016232          JMP     6$              ;GO TO 6$ IF ERROR
609 016216                  5$:
610 016216  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
    016222  000403                  BR      6$              ;GO TO 6$ IF NO ERROR
    016224  000240                  NOP                     ;RETURN HERE IF ERROR
    016226  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
    016230  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
611 016232                  6$:
612
613                         ;;*******************************************************************
                            ;*TEST 20        NOP TEST
                            ;;*******************************************************************
    016232                  TST20:
    016232  000004                  SCOPE                   ;SCOPE CALL
    016234  000240                  NOP                     ;START OF TEST
    016236  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
    016242  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
    016246  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
    016252  012737  000020  001226  MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
614
615 016260  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
    016264  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                            ;CLEAR CONTROLLER & SELECT DEVICE
                                                            ;VERIFY CONTROLLER CLEAR OPERATION
                                                            ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                            ;VERIFY PACK ACKNOWLEDGE
                                                            ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                            ;VERIFY RECALIBRATION
    016266  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
    016270  000240                  NOP                     ;RETURN HERE IF ERROR
    016272  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
    016274  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
616 016300                  1$:
617 016300  112737  000000  001545  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
    016306  112737  000200  001546  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
    016314  012737  000001  001412  MOV     #NOP!GO,RMCS10  ;SET RMCS1 OUTPUT BUFFER = NOP!GO
    016322  004737  043016          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
    016326  000404                  BR      2$              ;GO TO 2$ IF NO ERROR
    016330  000240                  NOP                     ;RETURN HERE IF ERROR
    016332  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
    016334  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
618 016340  004737  042462  2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
619
620 016344  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
    016350  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
```

```
                016352  000240                  NOP                     ;RETURN HERE IF ERROR
                016354  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                016356  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
621             016362                  3$:
622             016362  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                016366  000405                  BR      4$              ;GO TO 4$ IF NO ERROR
                016370  000240                  NOP                     ;RETURN HERE IF ERROR
                016372  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                016374  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                016376  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
623             016402                  4$:
624             016402  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                016406  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
                016410  000010                  .WORD   DPE             ;MASK FOR RMER2
                016412  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
                016414  000240                  NOP                     ;RETURN HERE IF ERROR
                016416  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                016420  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                016422  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
625             016426                  5$:
626             016426  004737  060210          JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
                016432  000405                  BR      6$              ;GO TO 6$ IF NO ERROR
                016434  000240                  NOP                     ;RETURN HERE IF ERROR
                016436  104000                  EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
                016440  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                016442  000137  016462          JMP     7$              ;GO TO 7$ IF ERROR
627             016446                  6$:
628             016446  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                016452  000403                  BR      7$              ;GO TO 7$ IF NO ERROR
                016454  000240                  NOP                     ;RETURN HERE IF ERROR
                016456  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                016460  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
629             016462                  7$:
630
631                                     ;;**************************************************************************
                                        ;*TEST 21        OFFSET TEST
                                        ;;**************************************************************************
                016462                  TST21:
                016462  000004                  SCOPE                   ;SCOPE CALL
                016464  000240                  NOP                     ;START OF TEST
                016466  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                016472  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                016476  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                016502  012737  000021  001226  MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
632
633             016510  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                016514  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
                016516  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                016520  000240                  NOP                     ;RETURN HERE IF ERROR
                016522  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                016524  000137  017006          JMP     9$              ;GO TO 9$ IF ERROR
```

```
634 016530                              1$:
635 016530  112737  000000  001545          MOVB    #RMCS1,PUTINX      ;SETUP PUT INDEX TABLE
    016536  112737  000200  001546          MOVB    #200,PUTINX+1      ;SET TERMINATOR BYTE
    016544  012737  000015  001412          MOV     #OFFSET!GO,RMCS10        ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
    016552  004737  043016                  JSR     PC,PUT             ;GO WRITE RMCS1 VIA PUT SUBROUTINE
    016556  000404                          BR      2$                 ;GO TO 2$ IF NO ERROR
    016560  000240                          NOP                        ;RETURN HERE IF ERROR
    016562  104000                          EMT                        ;ERROR DEFINED BY PUT SUBROUTINE
    016564  000137  017006                  JMP     9$                 ;GO TO 9$ IF ERROR
636 016570  004737  042462          2$:     JSR     PC,GETSTS          ;SETUP FOR STATUS FETCH
637 016574  004737  043370                  JSR     PC,TIMOUT          ;WAIT FOR GO TO RESET
638
639 016600  004737  042546                  JSR     PC,GET             ;GO READ REGISTER(S) WITH GET SUBROUTINE
    016604  000404                          BR      3$                 ;GO TO 3$ IF NO ERROR
    016606  000240                          NOP                        ;RETURN HERE IF ERROR
    016610  104000                          EMT                        ;ERROR # DEFINED BY GET SUBROUTINE
    016612  000137  017006                  JMP     9$                 ;GO TO 9$ IF ERROR
640 016616                              3$:
641 016616  004737  043564                  JSR     PC,PRIERR          ;GO CHECK FOR PRIMARY ERRORS
    016622  000405                          BR      4$                 ;GO TO 4$ IF NO ERROR
    016624  000240                          NOP                        ;RETURN HERE IF ERROR
    016626  104000                          EMT                        ;ERROR # DEFINED BY PRIERR SUBROUTINE
    016630  004736                          JSR     PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
    016632  000137  017006                  JMP     9$                 ;GO TO 9$ IF ERROR
642 016636  032737  000001  001350  4$:     BIT     #OM,RMDSI          ;OFFSET MODE ON??
643 016644  001012                          BNE     5$                 ;YES!!
644 016646  013737  001350  001142          MOV     RMDSI,$BDDAT       ;BAD DATA FOR TYPEOUT
645 016654  042737  177776  001142          BIC     #^COM,$BDDAT
646 016662  012737  000001  001140          MOV     #OM,$GDDAT         ;GOOD DATA FOR TYPEOUT
647 016670  104156                          EMT     156
648 016672  032737  100000  001350  5$:     BIT     #ATA,RMDSI         ;WAS ATTENTION SET ??
649 016700  001012                          BNE     6$                 ;YES!!
650 016702  013737  001350  001142          MOV     RMDSI,$BDDAT       ;BAD DATA FOR TYPEOUT
651 016710  042737  077777  001142          BIC     #^CATA,$BDDAT
652 016716  012737  100000  001140          MOV     #ATA,$GDDAT        ;GOOD DATA FOR TYPEOUT
653 016724  104172                          EMT     172
654 016726                              6$:
655 016726  004737  050562                  JSR     PC,CMPERRSTS       ;CHECK ANY ERRORS NOT MASKED
    016732  115760                          .WORD   NDTMSK             ;MASK FOR RMER1
    016734  000010                          .WORD   DPE                ;MASK FOR RMER2
    016736  000405                          BR      7$                 ;GO TO 7$ IF NO ERROR
    016740  000240                          NOP                        ;RETURN HERE IF ERROR
    016742  104000                          EMT                        ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
    016744  004736                          JSR     PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
    016746  000137  017006                  JMP     9$                 ;GO TO 9$ IF ERROR
656 016752                              7$:
657 016752  004737  060210                  JSR     PC,STCDRVSTS       ;GO CHECK FOR CHANGES IN DRIVE STATUS
    016756  000405                          BR      8$                 ;GO TO 8$ IF NO ERROR
    016760  000240                          NOP                        ;RETURN HERE IF ERROR
    016762  104000                          EMT                        ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
    016764  004736                          JSR     PC,@(SP)+          ;GO BACK FOR MORE ERROR CHECKS
    016766  000137  017006                  JMP     9$                 ;GO TO 9$ IF ERROR
658 016772                              8$:
659 016772  004737  044416                  JSR     PC,SECERR          ;GO CHECK FOR SECONDARY ERRORS
    016776  000403                          BR      9$                 ;GO TO 9$ IF NO ERROR
    017000  000240                          NOP                        ;RETURN HERE IF ERROR
    017002  104000                          EMT                        ;ERROR # DEFINED BY SECERR SUBROUTINE
```

```
                017004  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
660             017006                  9$:
661
662                                     ;;*****************************************************************
                                        ;*TEST 22        GO/ATA TEST
                                        ;;*****************************************************************
                017006                  TST22:
                017006  000004                  SCOPE                   ;SCOPE CALL
                017010  000240                  NOP                     ;START OF TEST
                017012  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                017016  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                017022  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                017026  012737  000022  001226  MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
663
664             017034  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                017040  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
                017042  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                017044  000240                  NOP                     ;RETURN HERE IF ERROR
                017046  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                017050  000137  017426          JMP     12$             ;GO TO 12$ IF ERROR
665             017054                  1$:
666             017054  112737  000000  001545  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                017062  112737  000200  001546  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                017070  012737  000015  001412  MOV     #OFFSET!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
                017076  004737  043016          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                017102  000404                  BR      2$              ;GO TO 2$ IF NO ERROR
                017104  000240                  NOP                     ;RETURN HERE IF ERROR
                017106  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                017110  000137  017426          JMP     12$             ;GO TO 12$ IF ERROR
667             017114  004737  042462  2$:     JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
668
669             017120  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                017124  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
                017126  000240                  NOP                     ;RETURN HERE IF ERROR
                017130  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                017132  000137  017426          JMP     12$             ;GO TO 12$ IF ERROR
670             017136  032737  100000  001350  3$:     BIT     #ATA,RMDSI      ;IS ATTENTION SET??
671             017144  001012                  BNE     4$              ;YES!!
672             017146  012737  100000  001140  MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
673             017154  013737  001350  001142  MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
674             017162  042737  077777  001142  BIC     #^CATA,$BDDAT
675             017170  104172                  EMT     172
676             017172                  4$:
677             017172  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                017176  000000                  .WORD   0               ;MASK FOR RMER1
                017200  000000                  .WORD   0               ;MASK FOR RMER2
                017202  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
                017204  000240                  NOP                     ;RETURN HERE IF ERROR
                017206  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                017210  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                017212  000137  017426          JMP     12$             ;GO TO 12$ IF ERROR
```

```
678 017216                          5$:
679 017216 112737 000000 001545     MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
    017224 112737 000200 001546     MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
    017232 012737 000001 001412     MOV     #NOP!GO,RMCS10  ;SET RMCS1 OUTPUT BUFFER = NOP!GO
    017240 004737 043016            JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
    017244 000404                   BR      6$              ;GO TO 6$ IF NO ERROR
    017246 000240                   NOP                     ;RETURN HERE IF ERROR
    017250 104000                   EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
    017252 000137 017426            JMP     12$             ;GO TO 12$ IF ERROR
680 017256                          6$:
681 017256 004737 042546            JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
    017262 000404                   BR      7$              ;GO TO 7$ IF NO ERROR
    017264 000240                   NOP                     ;RETURN HERE IF ERROR
    017266 104000                   EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
    017270 000137 017426            JMP     12$             ;GO TO 12$ IF ERROR
682 017274                          7$:
683 017274 004737 043564            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
    017300 000405                   BR      8$              ;GO TO 8$ IF NO ERROR
    017302 000240                   NOP                     ;RETURN HERE IF ERROR
    017304 104000                   EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
    017306 004736                   JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    017310 000137 017426            JMP     12$             ;GO TO 12$ IF ERROR
684 017314 032737 100000 001350 8$: BIT     #ATA,RMDSI      ;IS ATTENTION RESET??
685 017322 001411                   BEQ     9$              ;YES
686 017324 013737 001350 001142     MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
687 017332 042737 077777 001142     BIC     #^CATA,$BDDAT
688 017340 005037 001140            CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
689 017344 104246                   EMT     246
690 017346                          9$:
691 017346 004737 050562            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
    017352 115760                   .WORD   NDTMSK          ;MASK FOR RMER1
    017354 000010                   .WORD   DPE             ;MASK FOR RMER2
    017356 000405                   BR      10$             ;GO TO 10$ IF NO ERROR
    017360 000240                   NOP                     ;RETURN HERE IF ERROR
    017362 104000                   EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
    017364 004736                   JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    017366 000137 017426            JMP     12$             ;GO TO 12$ IF ERROR
692 017372                          10$:
693 017372 004737 060210            JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
    017376 000405                   BR      11$             ;GO TO 11$ IF NO ERROR
    017400 000240                   NOP                     ;RETURN HERE IF ERROR
    017402 104000                   EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
    017404 004736                   JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    017406 000137 017426            JMP     12$             ;GO TO 12$ IF ERROR
694 017412                          11$:
695 017412 004737 044416            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
    017416 000403                   BR      12$             ;GO TO 12$ IF NO ERROR
    017420 000240                   NOP                     ;RETURN HERE IF ERROR
    017422 104000                   EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
    017424 004736                   JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
696 017426                          12$:
697
698
                                    ;;*******************************************************************
                                    ;*TEST 23        WRITE ATA TEST
                                    ;;*******************************************************************
    017426                          TST23:
    017426 000004                   SCOPE                   ;SCOPE CALL
```

T23 WRITE ATA TEST

```
                017430  000240                          NOP             ;START OF TEST
                017432  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                017436  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                017442  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                017446  012737  000023  001226          MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    699
    700         017454  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                017460  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                017462  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                017464  000240                          NOP                     ;RETURN HERE IF ERROR
                017466  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                017470  000137  017772                  JMP     9$              ;GO TO 9$ IF ERROR
    701         017474                          1$:
    702         017474  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                017502  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                017510  012737  000015  001412          MOV     #OFFSET!GO,RMCS1O       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
                017516  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                017522  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                017524  000240                          NOP                     ;RETURN HERE IF ERROR
                017526  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                017530  000137  017772                  JMP     9$              ;GO TO 9$ IF ERROR
    703         017534  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
    704
    705         017540  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                017544  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                017546  000240                          NOP                     ;RETURN HERE IF ERROR
                017550  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                017552  000137  017772                  JMP     9$              ;GO TO 9$ IF ERROR
    706         017556  032737  100000  001350  3$:     BIT     #ATA,RMDSI      ;IS ATTENTION SET??
    707         017564  001012                          BNE     4$              ;YES!!
    708         017566  013737  001350  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
    709         017574  042737  077777  001142          BIC     #^CATA,$BDDAT
    710         017602  012737  100000  001140          MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
    711         017610  104172                          EMT     172
    712         017612  116102  000001          4$:     MOVB    1(R1),R2        ;R2 = ATTENTION BIT
    713         017616  042702  177400                  BIC     #^CATNMSK,R2    ;CLEAR SIGN EXTENSION
    714         017622  000240                          NOP
    715         017624  010237  001430                  MOV     R2,RMASO        ;PUT RMAS BIT IN OUTPUT BUF
    716         017630  112737  000016  001545          MOVB    #RMAS,PUTINX    ;WRITE REGISTER INDEX IN TABLE
    717         017636  112737  000200  001546          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR
    718         017644  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                017650  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                017652  000240                          NOP                     ;RETURN HERE IF ERROR
                017654  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                017656  000137  017772                  JMP     9$              ;GO TO 9$ IF ERROR
    719         017662                          5$:
    720         017662  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                017666  000404                          BR      6$              ;GO TO 6$ IF NO ERROR
                017670  000240                          NOP                     ;RETURN HERE IF ERROR
                017672  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                017674  000137  017772                  JMP     9$              ;GO TO 9$ IF ERROR
```

```
                                6$:
017700                                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
017700  004737  043564                  BR      7$              ;GO TO 7$ IF NO ERROR
017704  000405                          NOP                     ;RETURN HERE IF ERROR
017706  000240                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
017710  104000                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
017712  004736                          JMP     9$              ;GO TO 9$ IF ERROR
017714  000137  017772          7$:     BIT     #ATA,RMDSI      ;IS ATTENTION RESET??
017720  032737  100000  001350          BEQ     8$              ;YES!!
017726  001411                          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
017730  013737  001350  001142          BIC     #^CATA,$BDDAT
017736  042737  077777  001142          CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
017744  005037  001140                  EMT     247
017750  104247                  8$:
017752                          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
017752  004737  050562                  .WORD   0               ;MASK FOR RMER1
017756  000000                          .WORD   0               ;MASK FOR RMER2
017760  000000                          BR      9$              ;GO TO 9$ IF NO ERROR
017762  000403                          NOP                     ;RETURN HERE IF ERROR
017764  000240                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
017766  104000                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
017770  004736                  9$:
017772
                                ;;*******************************************************************
                                ;*TEST 24        ERROR/ATA TEST
                                ;;*******************************************************************
017772                          TST24:
017772  000004                          SCOPE                   ;SCOPE CALL
017774  000240                          NOP                     ;START OF TEST
017776  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
020002  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
020006  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
020012  012737  000024  001226          MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

020020  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
020024  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
020026  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
020030  000240                          NOP                     ;RETURN HERE IF ERROR
020032  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
020034  000137  020364                  JMP     10$             ;GO TO 10$ IF ERROR
020040                          1$:
020040  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
020046  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
020054  012737  000015  001412          MOV     #OFFSET!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
020062  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
020066  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
020070  000240                          NOP                     ;RETURN HERE IF ERROR
020072  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
020074  000137  020364                  JMP     10$             ;GO TO 10$ IF ERROR
020100  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
```

```
                                                ;GO READ REGISTER(S) WITH GET SUBROUTINE
740 020104 004737 042546        JSR     PC,GET
    020110 000404               BR      3$      ;GO TO 3$ IF NO ERROR
    020112 000240               NOP             ;RETURN HERE IF ERROR
    020114 104000               EMT             ;ERROR # DEFINED BY GET SUBROUTINE
    020116 000137 020364        JMP     10$     ;GO TO 10$ IF ERROR
741 020122 032737 100000 001350 3$:     BIT     #ATA,RMDSI      ;IS ATA SET??
742 020130 001012               BNE     4$      ;YES!!
743 020132 013737 001350 001142         MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
744 020140 042737 077777 001142         BIC     #^CATA,$BDDAT
745 020146 012737 100000 001140         MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
746 020154 104172               EMT     172
747 020156                      4$:
748 020156 112737 000014 001545         MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
    020164 112737 000200 001546         MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
    020172 012737 040000 001426         MOV     #UNS,RMER10     ;SET RMER1 OUTPUT BUFFER = UNS
    020200 004737 043016                JSR     PC,PUT          ;GO WRITE RMER1 VIA PUT SUBROUTINE
    020204 000404               BR      5$      ;GO TO 5$ IF NO ERROR
    020206 000240               NOP             ;RETURN HERE IF ERROR
    020210 104000               EMT             ;ERROR DEFINED BY PUT SUBROUTINE
    020212 000137 020364        JMP     10$     ;GO TO 10$ IF ERROR
749 020216                      5$:
750 020216 112737 000000 001545         MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
    020224 112737 000200 001546         MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
    020232 012737 000001 001412         MOV     #NOP!GO,RMCS10  ;SET RMCS1 OUTPUT BUFFER = NOP!GO
    020240 004737 043016                JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
    020244 000404               BR      6$      ;GO TO 6$ IF NO ERROR
    020246 000240               NOP             ;RETURN HERE IF ERROR
    020250 104000               EMT             ;ERROR DEFINED BY PUT SUBROUTINE
    020252 000137 020364        JMP     10$     ;GO TO 10$ IF ERROR
751 020256                      6$:
                                                ;GO READ REGISTER(S) WITH GET SUBROUTINE
752 020256 004737 042546        JSR     PC,GET
    020262 000404               BR      7$      ;GO TO 7$ IF NO ERROR
    020264 000240               NOP             ;RETURN HERE IF ERROR
    020266 104000               EMT             ;ERROR # DEFINED BY GET SUBROUTINE
    020270 000137 020364        JMP     10$     ;GO TO 10$ IF ERROR
753 020274 032737 100000 001350 7$:     BIT     #ATA,RMDSI      ;IS ATA STILL SET??
754 020302 001012               BNE     8$      ;YES!!
755 020304 012737 100000 001140         MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
756 020312 013737 001350 001142         MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
757 020320 042737 077777 001142         BIC     #^CATA,$BDDAT
758 020326 104250               EMT     250
759 020330                      8$:
760 020330 004737 060210                JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
    020334 000405               BR      9$      ;GO TO 9$ IF NO ERROR
    020336 000240               NOP             ;RETURN HERE IF ERROR
    020340 104000               EMT             ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
    020342 004736               JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    020344 000137 020364        JMP     10$     ;GO TO 10$ IF ERROR
761 020350                      9$:
762 020350 004737 044416                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
    020354 000403               BR      10$     ;GO TO 10$ IF NO ERROR
    020356 000240               NOP             ;RETURN HERE IF ERROR
    020360 104000               EMT             ;ERROR # DEFINED BY SECERR SUBROUTINE
    020362 004736               JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
763 020364                      10$:
764
765                             ;;*****************************************************************
```

```
                                        ;*TEST 25       PROGRAM INTERRUPT TEST
                                        ;;*****************************************************************
020364                                  TST25:
020364  000004                                  SCOPE                   ;SCOPE CALL
020366  000240                                  NOP                     ;START OF TEST
020370  012706  001100                          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
020374  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
020400  013701  001466                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
020404  012737  000025  001226                  MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
766
767 020412  004737  041532                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
    020416  054130                              .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
    020420  000404                              BR      1$              ;GO TO 1$ IF NO ERROR
    020422  000240                              NOP                     ;RETURN HERE IF ERROR
    020424  104000                              EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
    020426  000137  021054                      JMP     16$             ;GO TO 16$ IF ERROR
768 020432                              1$:
769 020432  113702  001272                      MOVB    $VECT1,R2       ;R2 = INTERRUPT ADDRESS
770 020436  042702  177400                      BIC     #^C<377>,R2     ;CLEAR SIGN EXTENSION
771 020442  011204                              MOV     (R2),R4         ;SAVE HANDLER ADDRESS
772 020444  016205  000002                      MOV     2(R2),R5        ;SAVE HANDLER PRIORITY
773 020450  012712  020714                      MOV     #9$,(R2)        ;WRITE HANDLER ADDRESS AND
774 020454  012762  000300  000002              MOV     #PR6,2(R2)      ;PRIORITY FOR THIS TEST
775 020462  113703  001273                      MOVB    $VECT1+1,R3     ;R3 = INTERRUPT LEVEL
776 020466  042703  177400                      BIC     #^C<377>,R3     ;CLEAR SIGN EXTENSION
777 020472  000303                              SWAB    R3
778 020474  005303                              DEC     R3              ;DECREMENT INTERRUPT LEVEL
779 020476  100001                              BPL     2$
780 020500  005003                              CLR     R3
781 020502  042703  177437              2$:     BIC     #^CPR7,R3
782 020506  000240                              NOP
783 020510  012746  000300                      MOV     #PR6,-(SP)      ;;PUSH #PR6 ON STACK
784 020514  012746  020652                      MOV     #6$,-(SP)       ;;PUSH #6$ ON STACK
785 020520  010346                              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
786 020522  012746  020644                      MOV     #5$,-(SP)       ;;PUSH #5$ ON STACK
787 020526  012737  177777  001430              MOV     #-1,RMASO       ;WRITE ONES IN RMAS TO
788 020534  112737  000016  001545              MOVB    #RMAS,PUTINX    ;CLEAR ALL ATTENTIONS
789 020542  112737  000200  001545              MOVB    #200,PUTINX
790
791 020550  004737  043016                      JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
    020554  000404                              BR      3$              ;GO TO 3$ IF NO ERROR
    020556  000240                              NOP                     ;RETURN HERE IF ERROR
    020560  104000                              EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
    020562  000137  021054                      JMP     16$             ;GO TO 16$ IF ERROR
792 020566                              3$:
793 020566  112737  000000  001545              MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
    020574  112737  000200  001546              MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
    020602  012737  000115  001412              MOV     #OFFSET!GO!IE,RMCS10    ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
    020610  004737  043016                      JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
    020614  000404                              BR      4$              ;GO TO 4$ IF NO ERROR
    020616  000240                              NOP                     ;RETURN HERE IF ERROR
```

```
020620  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
020622  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
794 020626                  4$:
795 020626  004737  043370      JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
796 020632  004737  042462      JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
797 020636  012703  000205      MOV     #205,R3         ;R3 = GROSS TIMEOUT
798 020642  000002              RTI                     ;DROP CP PRIORITY
799 020644  005303          5$: DEC     R3              ;TIMEOUT THE INTERRUPT
800 020646  100376              BPL     5$
801
802                         ;TIMEOUT BEFORE INTERRUPT
803
804 020650  000002              RTI                     ;RAISE CP PRIORITY
805 020652                  6$:
806 020652  004737  042546      JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
020656  000404                  BR      7$              ;GO TO 7$ IF NO ERROR
020660  000240                  NOP                     ;RETURN HERE IF ERROR
020662  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
020664  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
807 020670                  7$:
808 020670  004737  043564      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
020674  000405                  BR      8$              ;GO TO 8$ IF NO ERROR
020676  000240                  NOP                     ;RETURN HERE IF ERROR
020700  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
020702  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020704  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
809 020710                  8$:
020710  104251                  EMT     251
810 020712  000423              BR      12$
811
812 020714  022626          9$: CMP     (SP)+,(SP)+     ;ADJUST STACK
813 020716  012716  020724      MOV     #10$,(SP)       ;CHANGE RETURN ADDRESS
814 020722  000002              RTI                     ;RAISE CP PRIORITY
815 020724                  10$:
816 020724  004737  042546      JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
020730  000404                  BR      11$             ;GO TO 11$ IF NO ERROR
020732  000240                  NOP                     ;RETURN HERE IF ERROR
020734  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
020736  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
817 020742                  11$:
818 020742  004737  043564      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
020746  000405                  BR      12$             ;GO TO 12$ IF NO ERROR
020750  000240                  NOP                     ;RETURN HERE IF ERROR
020752  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
020754  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020756  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
819 020762                  12$:
820 020762  004737  050562      JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
020766  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
020770  000010                  .WORD   DPE             ;MASK FOR RMER2
020772  000405                  BR      13$             ;GO TO 13$ IF NO ERROR
020774  000240                  NOP                     ;RETURN HERE IF ERROR
020776  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
021000  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021002  000137  021054          JMP     16$             ;GO TO 16$ IF ERROR
821 021006                  13$:
822 021006  004737  060210      JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
```

```
021012  000405                          BR      14$             ;GO TO 14$ IF NO ERROR
021014  000240                          NOP                     ;RETURN HERE IF ERROR
021016  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
021020  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021022  000137  021054                  JMP     16$             ;GO TO 16$ IF ERROR
823 021026                      14$:
824 021026  004737  044416              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
021032  000405                          BR      15$             ;GO TO 15$ IF NO ERROR
021034  000240                          NOP                     ;RETURN HERE IF ERROR
021036  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
021040  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021042  000137  021054                  JMP     16$             ;GO TO 16$ IF ERROR
825 021046                      15$:
826 021046  010412                      MOV     R4,(R2)
827 021050  010562  000002              MOV     R5,2(R2)
828 021054                      16$:
829
830                             ;;*******************************************************************
                                ;*TEST 26        INHIBIT INTERRUPT TEST
                                ;;*******************************************************************
021054                          TST26:
021054  000004                          SCOPE                   ;SCOPE CALL
021056  000240                          NOP                     ;START OF TEST
021060  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
021064  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
021070  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
021074  012737  000026  001226          MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
831
832 021102  004737  041532              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
021106  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
021110  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
021112  000240                          NOP                     ;RETURN HERE IF ERROR
021114  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
021116  000137  021422                  JMP     13$             ;GO TO 13$ IF ERROR
833 021122                      1$:
834 021122  113702  001272              MOVB    $VECT1,R2       ;R2 = INTERRUPT ADDRESS
835 021126  042702  177400              BIC     #^C<377>,R2     ;CLEAR SIGN EXTENSION
836 021132  011204                      MOV     (R2),R4         ;SAVE HANDLER ADDRESS
837 021134  016205  000002              MOV     2(R2),R5        ;SAVE HANDLER PRIORITY
838 021140  012712  021362              MOV     #8$,(R2)        ;WRITE HANDLER ADDRESS AND
839 021144  012762  000300  000002      MOV     #PR6,2(R2)      ;PRIORITY FOR THIS TEST
840 021152  113703  001273              MOVB    $VECT1+1,R3     ;R3 = INTERRUPT LEVEL
841 021156  042703  177400              BIC     #^C<377>,R3     ;CLEAR SIGN EXTENSION
842 021162  012746  000300              MOV     #PR6,-(SP)      ;;PUSH #PR6 ON STACK
843 021166  012746  021414              MOV     #12$,-(SP)      ;;PUSH #12$ ON STACK
844 021172  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
845 021174  005316                      DEC     (SP)
846 021176  100001                      BPL     2$
847 021200  005016                      CLR     (SP)
848 021202  042716  177437      2$:     BIC     #^CPR7,(SP)
849 021206  012746  021350              MOV     #6$,-(SP)       ;;PUSH #6$ ON STACK
```

```
                                                MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                                MOV     #4$,-(SP)       ;;PUSH #4$ ON STACK
021220  004737  042462                          JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS

021224  112737  000000  001545                  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
021232  112737  000200  001546                  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
021240  012737  000115  001412                  MOV     #OFFSET!GO!IE,RMCS10    ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO!IE
021246  004737  043016                          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
021252  000404                                  BR      3$              ;GO TO 3$ IF NO ERROR
021254  000240                                  NOP                     ;RETURN HERE IF ERROR
021256  104000                                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
021260  000137  021422                          JMP     13$             ;GO TO 13$ IF ERROR
021264                          3$:
021264  004737  043370                          JSR     PC,TIMOUT       ;WAIT FOR GO = 0
021270  012703  000205                          MOV     #205,R3         ;R3 = GROSS TIMER
021274  000002                                  RTI

021276  005303                          4$:     DEC     R3              ;DELAY
021300  100376                                  BPL     4$

021302  112737  000000  001545                  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
021310  112737  000200  001546                  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
021316  012737  000000  001412                  MOV     #NOP,RMCS10     ;SET RMCS1 OUTPUT BUFFER = NOP
021324  004737  043016                          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
021330  000404                                  BR      5$              ;GO TO 5$ IF NO ERROR
021332  000240                                  NOP                     ;RETURN HERE IF ERROR
021334  104000                                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
021336  000137  021422                          JMP     13$             ;GO TO 13$ IF ERROR
021342  012703  000205                  5$:     MOV     #205,R3
021346  000002                                  RTI

021350  012712  021364                  6$:     MOV     #9$,(R2)
021354  005303                          7$:     DEC     R3
021356  100376                                  BPL     7$
021360  000002                                  RTI                             ;NO ERROR

021362  022626                          8$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
021364  022626                          9$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
021366  012716  021374                          MOV     #10$,(SP)
021372  000002                                  RTI
021374                          10$:
021374  004737  042546                          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
021400  000404                                  BR      11$             ;GO TO 11$ IF NO ERROR
021402  000240                                  NOP                     ;RETURN HERE IF ERROR
021404  104000                                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
021406  000137  021422                          JMP     13$             ;GO TO 13$ IF ERROR
021412                          11$:
021412  104252                                  EMT     252

021414  010412                          12$:    MOV     R4,(R2)
021416  010562  000002                          MOV     R5,2(R2)
021422                          13$:


                        ;;**************************************************************************
                        ;*TEST 27        RETURN TO CENTERLINE TEST
                        ;;**************************************************************************
021422                  TST27:
```

```
                021422  000004                          SCOPE                   ;SCOPE CALL
                021424  000240                          NOP                     ;START OF TEST
                021426  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                021432  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                021436  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                021442  012737  000027  001226          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    885
    886         021450  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                021454  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                021456  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                021460  000240                          NOP                     ;RETURN HERE IF ERROR
                021462  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                021464  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
    887         021470                          1$:
    888         021470  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                021476  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                021504  012737  000017  001412          MOV     #RTC!GO,RMCS1O  ;SET RMCS1 OUTPUT BUFFER = RTC!GO
                021512  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                021516  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                021520  000240                          NOP                     ;RETURN HERE IF ERROR
                021522  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                021524  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
    889         021530  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
    890         021534  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR RECAL TO COMPLETE
    891
    892         021540  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                021544  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                021546  000240                          NOP                     ;RETURN HERE IF ERROR
                021550  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                021552  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
    893         021556                          3$:
    894         021556  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                021562  000405                          BR      4$              ;GO TO 4$ IF NO ERROR
                021564  000240                          NOP                     ;RETURN HERE IF ERROR
                021566  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                021570  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                021572  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
    895         021576  032737  000001  001350  4$:     BIT     #OM,RMDSI       ;OFFSET MODE OFF??
    896         021604  001411                          BEQ     5$              ;YES!!
    897         021606  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    898         021612  013737  001350  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
    899         021620  042737  177776  001142          BIC     #^COM,$BDDAT
    900         021626  104157                          EMT     157
    901         021630  032737  100000  001350  5$:     BIT     #ATA,RMDSI      ;WAS ATTENTION SET ??
    902         021636  001012                          BNE     6$              ;YES!!
    903         021640  013737  001350  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
    904         021646  042737  077777  001142          BIC     #^CATA,$BDDAT
    905         021654  012737  100000  001140          MOV     #ATA,$GDDAT     ;GOOD DATA FOR TYPEOUT
    906         021662  104171                          EMT     171
    907         021664                          6$:
    908         021664  004737  050562                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
```

```
021670  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
021672  000010                          .WORD   DPE             ;MASK FOR RMER2
021674  000405                          BR      7$              ;GO TO 7$ IF NO ERROR
021676  000240                          NOP                     ;RETURN HERE IF ERROR
021700  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
021702  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021704  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
909 021710                      7$:
910 021710  004737  060210              JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
021714  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
021716  000240                          NOP                     ;RETURN HERE IF ERROR
021720  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
021722  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021724  000137  021744                  JMP     9$              ;GO TO 9$ IF ERROR
911 021730                      8$:
912 021730  004737  044416              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
021734  000403                          BR      9$              ;GO TO 9$ IF NO ERROR
021736  000240                          NOP                     ;RETURN HERE IF ERROR
021740  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
021742  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
913 021744                      9$:
914
915                             ;;*******************************************************************
                                ;*TEST 30        READ IN PRESET TEST
                                ;;*******************************************************************
021744                          TST30:
021744  000004                          SCOPE                   ;SCOPE CALL
021746  000240                          NOP                     ;START OF TEST
021750  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
021754  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
021760  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
021764  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
916
917 021772  004737  041532              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
021776  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
022000  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
022002  000240                          NOP                     ;RETURN HERE IF ERROR
022004  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
022006  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
918 022012                      1$:
919 022012  012702  001545              MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
920 022016  112722  000006              MOVB    #RMDA,(R2)+
921 022022  112722  000034              MOVB    #RMDC,(R2)+
922 022026  112722  000032              MOVB    #RMOF,(R2)+
923 022032  112722  000000              MOVB    #RMCS1,(R2)+
924 022036  112722  000200              MOVB    #200,(R2)+
925 022042  012737  177777  001420      MOV     #-1,RMDA0
926 022050  012737  177777  001446      MOV     #-1,RMDC0
927 022056  012737  017200  001444      MOV     #FMT16!ECI!HCI!SSEI!OFD,RMOF0
928 022064  012737  000021  001412      MOV     #RIP!GO,RMCS10
929 022072  004737  043016              JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
```

```
                022076  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                022100  000240                          NOP                     ;RETURN HERE IF ERROR
                022102  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                022104  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
    930         022110  004737  042462          2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
    931         022114  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR RIP TO COMPLETE
    932
    933         022120  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                022124  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                022126  000240                          NOP                     ;RETURN HERE IF ERROR
                022130  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                022132  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
    934         022136                          3$:
    935         022136  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                022142  000405                          BR      4$              ;GO TO 4$ IF NO ERROR
                022144  000240                          NOP                     ;RETURN HERE IF ERROR
                022146  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                022150  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                022152  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
    936         022156  013737  001370  001142  4$:     MOV     RMOFI,$BDDAT    ;IS RMOF RESET??
    937         022164  042737  160577  001142          BIC     #^C<FMT16!ECI!HCI!SSEI!OFD>,$BDDAT
    938         022172  001403                          BEQ     5$              ;YES!!
    939         022174  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    940         022200  104160                          EMT     160
    941         022202  005737  001344          5$:     TST     RMDAI           ;IS RMDA RESET??
    942         022206  001406                          BEQ     6$              ;YES!!
    943         022210  013737  001344  001142          MOV     RMDAI,$BDDAT    ;BAD DATA FOR TYPEOUT
    944         022216  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    945         022222  104161                          EMT     161
    946         022224  005737  001372          6$:     TST     RMDCI           ;IS RMDC RESET??
    947         022230  001406                          BEQ     7$              ;YES!!
    948         022232  013737  001372  001142          MOV     RMDCI,$BDDAT    ;BAD DATA FOR TYPEOUT
    949         022240  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    950         022244  104162                          EMT     162
    951         022246                          7$:
    952         022246  004737  050562                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                022252  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
                022254  000010                          .WORD   DPE             ;MASK FOR RMER2
                022256  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
                022260  000240                          NOP                     ;RETURN HERE IF ERROR
                022262  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                022264  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                022266  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
    953         022272                          8$:
    954         022272  004737  060210                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
                022276  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
                022300  000240                          NOP                     ;RETURN HERE IF ERROR
                022302  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
                022304  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                022306  000137  022326                  JMP     10$             ;GO TO 10$ IF ERROR
    955         022312                          9$:
    956         022312  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                022316  000403                          BR      10$             ;GO TO 10$ IF NO ERROR
                022320  000240                          NOP                     ;RETURN HERE IF ERROR
                022322  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                022324  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
    957         022326                          10$:
```

```
958
959                                     ;;*****************************************************************
                                        ;*TEST 31        RMDC CLEAR OFFSET TEST
                                        ;;*****************************************************************
     022326                             TST31:
     022326  000004                             SCOPE                   ;SCOPE CALL
     022330  000240                             NOP                     ;START OF TEST
     022332  012706  001100                     MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     022336  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     022342  013701  001466                     MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     022346  012737  000031  001226             MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
960
961  022354  004737  041532                     JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     022360  054130                             .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                         ;CLEAR CONTROLLER & SELECT DEVICE
                                                                         ;VERIFY CONTROLLER CLEAR OPERATION
                                                                         ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                         ;VERIFY PACK ACKNOWLEDGE
                                                                         ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                         ;VERIFY RECALIBRATION
     022362  000404                             BR      1$              ;GO TO 1$ IF NO ERROR
     022364  000240                             NOP                     ;RETURN HERE IF ERROR
     022366  104000                             EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     022370  000137  022644                     JMP     8$              ;GO TO 8$ IF ERROR
962  022374                             1$:
963  022374  112737  000000  001545             MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
     022402  112737  000200  001546             MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     022410  012737  000015  001412             MOV     #OFFSET!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
     022416  004737  043016                     JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
     022422  000404                             BR      2$              ;GO TO 2$ IF NO ERROR
     022424  000240                             NOP                     ;RETURN HERE IF ERROR
     022426  104000                             EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     022430  000137  022644                     JMP     8$              ;GO TO 8$ IF ERROR
964  022434  004737  042462             2$:     JSR     PC,GETSTS       ;SETUP FOR STATUS
965
966  022440  004737  042546                     JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     022444  000404                             BR      3$              ;GO TO 3$ IF NO ERROR
     022446  000240                             NOP                     ;RETURN HERE IF ERROR
     022450  104000                             EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     022452  000137  022644                     JMP     8$              ;GO TO 8$ IF ERROR
967  022456                             3$:
968  022456  004737  043564                     JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     022462  000405                             BR      4$              ;GO TO 4$ IF NO ERROR
     022464  000240                             NOP                     ;RETURN HERE IF ERROR
     022466  104000                             EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     022470  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     022472  000137  022644                     JMP     8$              ;GO TO 8$ IF ERROR
969  022476  032737  000001  001350     4$:     BIT     #OM,RMDSI       ;OFFSET ON??
970  022504  001013                             BNE     5$              ;YES
971  022506  013737  001350  001142             MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
972  022514  042737  177776  001142             BIC     #^COM,$BDDAT
973  022522  012737  000001  001140             MOV     #OM,$GDDAT      ;GOOD DATA FOR TYPEOUT
974  022530  104156                             EMT     156
975  022532  000444                             BR      8$              ;SKIP REST OF TEST
976  022534                             5$:
977  022534  112737  000034  001545             MOVB    #RMDC,PUTINX    ;SETUP PUT INDEX TABLE
     022542  112737  000200  001546             MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
```

```
                022550  012737  000000  001446          MOV     #0,RMDCO        ;SET RMDC OUTPUT BUFFER = 0
                022556  004737  043016                  JSR     PC,PUT          ;GO WRITE RMDC VIA PUT SUBROUTINE
                022562  000404                          BR      6$              ;GO TO 6$ IF NO ERROR
                022564  000240                          NOP                     ;RETURN HERE IF ERROR
                022566  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                022570  000137  022644                  JMP     8$              ;GO TO 8$ IF ERROR
    978         022574                          6$:
    979         022574  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                022600  000404                          BR      7$              ;GO TO 7$ IF NO ERROR
                022602  000240                          NOP                     ;RETURN HERE IF ERROR
                022604  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                022606  000137  022644                  JMP     8$              ;GO TO 8$ IF ERROR
    980         022612  032737  000001  001350  7$:     BIT     #OM,RMDSI       ;DID OFFSET MODE RESET??
    981         022620  001411                          BEQ     8$
    982         022622  013737  001350  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
    983         022630  042737  177776  001142          BIC     #^COM,$BDDAT
    984         022636  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
    985         022642  104253                          EMT     253
    986         022644                          8$:
    987
    988                                         ;;*****************************************************************
                                                ;*TEST 32        ILLEGAL FUNCTION TEST
                                                ;;*****************************************************************
                022644                          TST32:
                022644  000004                          SCOPE                   ;SCOPE CALL
                022646  000240                          NOP                     ;START OF TEST
                022650  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                022654  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                022660  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                022664  012737  000032  001226          MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    989
    990         022672  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                022676  040000                          .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                022700  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                022702  000240                          NOP                     ;RETURN HERE IF ERROR
                022704  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                022706  000137  023430                  JMP     13$             ;GO TO 13$ IF ERROR
    991         022712                          1$:
    992         022712  004737  042462                  JSR     PC,GETSTS       ;SETUP FOR STATUS
    993         022716  012702  000000                  MOV     #NOP,R2
    994         022722                          2$:
    995         022722  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                022726  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                022730  000240                          NOP                     ;RETURN HERE IF ERROR
                022732  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                022734  000137  023430                  JMP     13$             ;GO TO 13$ IF ERROR
    996         022740                          3$:
    997
    998                                         ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
    999                                         ;CYLINDER
   1000         022740  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                022746  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                022754  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
                022762  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                022766  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                022770  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
                022772  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                022774  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
1001            023000                  4$:
1002            023000  112737  000024  001545  MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                023006  112737  000200  001546  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                023014  012737  001401  001436  MOV     #DMD!MUR!MOC,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC
                023022  004737  043016          JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                023026  000404                  BR      5$              ;GO TO 5$ IF NO ERROR
                023030  000240                  NOP                     ;RETURN HERE IF ERROR
                023032  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                023034  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
1003            023040                  5$:
1004
1005                                    ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
1006            023040  112737  000000  001545  MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                023046  112737  000200  001546  MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                023054  012737  000023  001412  MOV     #PAKACK!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
                023062  004737  043016          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                023066  000404                  BR      6$              ;GO TO 6$ IF NO ERROR
                023070  000240                  NOP                     ;RETURN HERE IF ERROR
                023072  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                023074  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
1007            023100                  6$:
1008            023100  012737  000001  001412  MOV     #GO,RMCS10
1009            023106  050237  001412          BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
1010            023112  012737  104372  001416  MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
1011            023120  012737  177777  001414  MOV     #-1,RMWC0       ;DUMMY WORD COUNT
1012            023126  005037  001420          CLR     RMDA0           ;CLEAR DISK ADDRESS
1013            023132  005037  001446          CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
1014            023136  012737  010000  001444  MOV     #FMT16,RMOF0    ;16 BIT FORMAT
1015            023144  012703  001545          MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
1016            023150  112723  000004          MOVB    #RMBA,(R3)+
1017            023154  112723  000002          MOVB    #RMWC,(R3)+
1018            023160  112723  000032          MOVB    #RMOF,(R3)+
1019            023164  112723  000006          MOVB    #RMDA,(R3)+
1020            023170  112723  000034          MOVB    #RMDC,(R3)+
1021            023174  112723  000000          MOVB    #RMCS1,(R3)+
1022            023200  112713  000200          MOVB    #200,(R3)
1023            023204  004737  043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                023210  000404                  BR      7$              ;GO TO 7$ IF NO ERROR
                023212  000240                  NOP                     ;RETURN HERE IF ERROR
                023214  104000                  EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                023216  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
1024            023222  004737  043370  7$:     JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
1025
1026            023226  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                023232  000404                  BR      8$              ;GO TO 8$ IF NO ERROR
                023234  000240                  NOP                     ;RETURN HERE IF ERROR
                023236  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                023240  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
1027            023244                  8$:
1028            023244  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                023250  000405                  BR      9$              ;GO TO 9$ IF NO ERROR
                023252  000240                  NOP                     ;RETURN HERE IF ERROR
                023254  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                023256  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                023260  000137  023430          JMP     13$             ;GO TO 13$ IF ERROR
```

```
1029 023264 016237 070002 001140 9$:    MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
1030 023272 042737 177776 001140        BIC     #^CILF,$GDDAT
1031 023300 013737 001352 001142        MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
1032 023306 042737 177776 001142        BIC     #^CILF,$BDDAT
1033 023314 023737 001140 001142        CMP     $GDDAT,$BDDAT   ;IS ILF STATUS CORRECT?
1034 023322 001402                      BEQ     10$             ;YES!!
1035 023324 104254                      EMT     254
1036 023326 000440                      BR      13$
1037 023330 016237 070002 001140 10$:   MOV     FNCDTB(R2),$GDDAT
1038 023336 032737 040000 001350        BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
1039 023344 001403                      BEQ     11$             ;NO!!
1040 023346 052737 100000 001140        BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
1041 023354 042737 077777 001140 11$:   BIC     #^CATA,$GDDAT
1042 023362 013737 001350 001142        MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
1043 023370 042737 077777 001142        BIC     #^CATA,$BDDAT
1044 023376 023737 001140 001142        CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
1045 023404 001402                      BEQ     12$             ;YES!!
1046 023406 104255                      EMT     255
1047 023410 000407                      BR      13$
1048 023412 062702 000002       12$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
1049 023416 022702 000076               CMP     #ILF76,R2       ;DONE??
1050 023422 103402                      BLO     13$             ;YES !!
1051 023424 000137 022722               JMP     2$              ;TEST NEXT FUNCTION
1052 023430                     13$:
1053
1054                            ;;*******************************************************************
                                ;*TEST 33        INVALID COMMAND TEST
                                ;;*******************************************************************
     023430                     TST33:
     023430 000004                      SCOPE                   ;SCOPE CALL
     023432 000240                      NOP                     ;START OF TEST
     023434 012706 001100               MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     023440 013700 001276               MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     023444 013701 001466               MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     023450 012737 000033 001226        MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1055
1056 023456 004737 041532               JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     023462 040000                      .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
     023464 000404                      BR      1$              ;GO TO 1$ IF NO ERROR
     023466 000240                      NOP                     ;RETURN HERE IF ERROR
     023470 104000                      EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     023472 000137 024154               JMP     13$             ;GO TO 13$ IF ERROR
1057 023476                     1$:
1058 023476 004737 042462               JSR     PC,GETSTS       ;SETUP FOR STATUS
1059 023502 012702 000000               MOV     #NOP,R2
1060 023506                     2$:
1061 023506 004737 052466               JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     023512 000404                      BR      3$              ;GO TO 3$ IF NO ERROR
     023514 000240                      NOP                     ;RETURN HERE IF ERROR
     023516 104000                      EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     023520 000137 024154               JMP     13$             ;GO TO 13$ IF ERROR
1062 023524                     3$:
1063
1064                            ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
1065                            ;CYLINDER
1066 023524 112737 000024 001545        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
```

```
                023532  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                023540  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
                023546  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                023552  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                023554  000240                          NOP                     ;RETURN HERE IF ERROR
                023556  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                023560  000137  024154                  JMP     13$             ;GO TO 13$ IF ERROR
     1067       023564                          4$:
     1068       023564  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                023572  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                023600  012737  001401  001436          MOV     #DMD!MUR!MOC,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC
                023606  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                023612  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                023614  000240                          NOP                     ;RETURN HERE IF ERROR
                023616  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                023620  000137  024154                  JMP     13$             ;GO TO 13$ IF ERROR
     1069       023624                          5$:
     1070
     1071
     1072                                       ;VOLUME VALID IS LEFT RESET FOR THE TEST
     1073
     1074       023624                          6$:
     1075       023624  012737  000001  001412          MOV     #GO,RMCS10
     1076       023632  050237  001412                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
     1077       023636  012737  104372  001416          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
     1078       023644  012737  177777  001414          MOV     #-1,RMWC0       ;DUMMY WORD COUNT
     1079       023652  005037  001420                  CLR     RMDA0           ;CLEAR DISK ADDRESS
     1080       023656  005037  001446                  CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
     1081       023662  012737  010000  001444          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
     1082       023670  012703  001545                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
     1083       023674  112723  000004                  MOVB    #RMBA,(R3)+
     1084       023700  112723  000002                  MOVB    #RMWC,(R3)+
     1085       023704  112723  000032                  MOVB    #RMOF,(R3)+
     1086       023710  112723  000006                  MOVB    #RMDA,(R3)+
     1087       023714  112723  000034                  MOVB    #RMDC,(R3)+
     1088       023720  112723  000000                  MOVB    #RMCS1,(R3)+
     1089       023724  112713  000200                  MOVB    #200,(R3)
     1090       023730  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                023734  000404                          BR      7$              ;GO TO 7$ IF NO ERROR
                023736  000240                          NOP                     ;RETURN HERE IF ERROR
                023740  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                023742  000137  024154                  JMP     13$             ;GO TO 13$ IF ERROR
     1091       023746  004737  043370          7$:     JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
     1092
     1093       023752  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                023756  000404                          BR      8$              ;GO TO 8$ IF NO ERROR
                023760  000240                          NOP                     ;RETURN HERE IF ERROR
                023762  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                023764  000137  024154                  JMP     13$             ;GO TO 13$ IF ERROR
     1094       023770                          8$:
     1095       023770  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                023774  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
                023776  000240                          NOP                     ;RETURN HERE IF ERROR
                024000  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                024002  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                024004  000137  024154                  JMP     13$             ;GO TO 13$ IF ERROR
     1096       024010  016237  070002  001140  9$:     MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
```

```
1097 024016 042737 167777 001140          BIC     #^CIVC,$GDDAT
1098 024024 013737 001400 001142          MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
1099 024032 042737 167777 001142          BIC     #^CIVC,$BDDAT
1100 024040 023737 001140 001142          CMP     $GDDAT,$BDDAT   ;IS IVC STATUS CORRECT?
1101 024046 001402                        BEQ     10$             ;YES!!
1102 024050 104216                        EMT     216
1103 024052 000440                        BR      13$
1104 024054 016237 070002 001140  10$:    MOV     FNCDTB(R2),$GDDAT
1105 024062 032737 040000 001350          BIT     #EKR,RMDSI      ;WAS AN ERROR DETECTED??
1106 024070 001403                        BEQ     11$             ;NO!!
1107 024072 052737 100000 001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
1108 024100 042737 077777 001140  11$:    BIC     #^CATA,$GDDAT
1109 024106 013737 001350 001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
1110 024114 042737 077777 001142          BIC     #^CATA,$BDDAT
1111 024122 023737 001140 001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
1112 024130 001402                        BEQ     12$             ;YES!!
1113 024132 104255                        EMT     255
1114 024134 000407                        BR      13$
1115 024136 062702 000002         12$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
1116 024142 022702 000076                 CMP     #ILF76,R2       ;DONE??
1117 024146 103402                        BLO     13$             ;YES !!
1118 024150 000137 023506                 JMP     2$              ;TEST NEXT FUNCTION
1119 024154               13$:
1120
1121                                      ;;*****************************************************************
                                          ;*TEST 34       INVALID ADDRESS ERROR TEST
                                          ;;*****************************************************************
     024154               TST34:
     024154 000004                        SCOPE                   ;SCOPE CALL
     024156 000240                        NOP                     ;START OF TEST
     024160 012706 001100                 MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     024164 013700 001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     024170 013701 001466                 MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     024174 012737 000034 001226          MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1122
1123 024202 004737 041532                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     024206 040000                        .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                  ;CLEAR CONTROLLER & SELECT DEVICE
     024210 000404                        BR      1$              ;GO TO 1$ IF NO ERROR
     024212 000240                        NOP                     ;RETURN HERE IF ERROR
     024214 104000                        EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     024216 000137 024744                 JMP     13$             ;GO TO 13$ IF ERROR
1124 024222               1$:
1125 024222 004737 042462                 JSR     PC,GETSTS       ;SETUP FOR STATUS
1126 024226 012702 000000                 MOV     #NOP,R2
1127 024232               2$:
1128 024232 004737 052466                 JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     024236 000404                        BR      3$              ;GO TO 3$ IF NO ERROR
     024240 000240                        NOP                     ;RETURN HERE IF ERROR
     024242 104000                        EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     024244 000137 024744                 JMP     13$             ;GO TO 13$ IF ERROR
1129 024250               3$:
1130
1131                                      ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
1132                                      ;CYLINDER
1133 024250 112737 000024 001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
     024256 112737 000200 001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
```

```
                024264  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
                024272  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                024276  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                024300  000240                          NOP                     ;RETURN HERE IF ERROR
                024302  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                024304  000137  024744                  JMP     13$             ;GO TO 13$ IF ERROR
     1134       024310                          4$:
     1135       024310  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                024316  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                024324  012737  001401  001436          MOV     #DMD!MUR!MOC,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC
                024332  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
                024336  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                024340  000240                          NOP                     ;RETURN HERE IF ERROR
                024342  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                024344  000137  024744                  JMP     13$             ;GO TO 13$ IF ERROR
     1136       024350                          5$:
     1137
     1138                                       ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
     1139       024350  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
                024356  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
                024364  012737  000023  001412          MOV     #PAKACK!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
                024372  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
                024376  000404                          BR      6$              ;GO TO 6$ IF NO ERROR
                024400  000240                          NOP                     ;RETURN HERE IF ERROR
                024402  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
                024404  000137  024744                  JMP     13$             ;GO TO 13$ IF ERROR
     1140       024410                          6$:
     1141       024410  012737  000001  001412          MOV     #GO,RMCS10
     1142       024416  050237  001412                  BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
     1143       024422  012737  104372  001416          MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
     1144       024430  012737  177777  001414          MOV     #-1,RMWC0       ;DUMMY WORD COUNT
     1145       024436  012737  177777  001420          MOV     #-1,RMDA0       ;USE INVALID DISK ADDRESS
     1146       024444  012737  177777  001446          MOV     #-1,RMDC0       ;USE INVALID CYLINDER ADDRESS
     1147       024452  012737  010000  001444          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
     1148       024460  012703  001545                  MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
     1149       024464  112723  000004                  MOVB    #RMBA,(R3)+
     1150       024470  112723  000002                  MOVB    #RMWC,(R3)+
     1151       024474  112723  000032                  MOVB    #RMOF,(R3)+
     1152       024500  112723  000006                  MOVB    #RMDA,(R3)+
     1153       024504  112723  000034                  MOVB    #RMDC,(R3)+
     1154       024510  112723  000000                  MOVB    #RMCS1,(R3)+
     1155       024514  112713  000200                  MOVB    #200,(R3)
     1156       024520  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                024524  000404                          BR      7$              ;GO TO 7$ IF NO ERROR
                024526  000240                          NOP                     ;RETURN HERE IF ERROR
                024530  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                024532  000137  024744                  JMP     13$             ;GO TO 13$ IF ERROR
     1157       024536  004737  043370          7$:     JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
     1158
     1159       024542  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                024546  000404                          BR      8$              ;GO TO 8$ IF NO ERROR
                024550  000240                          NOP                     ;RETURN HERE IF ERROR
                024552  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                024554  000137  024744                  JMP     13$             ;GO TO 13$ IF ERROR
     1160       024560                          8$:
     1161       024560  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                024564  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
```

```
                024566  000240                  NOP                     ;RETURN HERE IF ERROR
                024570  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                024572  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                024574  000137  024744          JMP     13$             ;GO TO 13$ IF ERROR
1162            024600  016237  070002  001140  9$:     MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
1163            024606  042737  175777  001140          BIC     #^CIAE,$GDDAT
1164            024614  013737  001352  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
1165            024622  042737  175777  001142          BIC     #^CIAE,$BDDAT
1166            024630  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS IAE STATUS CORRECT?
1167            024636  001402                          BEQ     10$             ;YES!!
1168            024640  104217                          EMT     217
1169            024642  000440                          BR      13$
1170            024644  016237  070002  001140  10$:    MOV     FNCDTB(R2),$GDDAT
1171            024652  032737  040000  001350          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
1172            024660  001403                          BEQ     11$             ;NO!!
1173            024662  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
1174            024670  042737  077777  001140  11$:    BIC     #^CATA,$GDDAT
1175            024676  013737  001350  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
1176            024704  042737  077777  001142          BIC     #^CATA,$BDDAT
1177            024712  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
1178            024720  001402                          BEQ     12$             ;YES!!
1179            024722  104255                          EMT     255
1180            024724  000407                          BR      13$
1181            024726  062702  000002          12$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
1182            024732  022702  000076                  CMP     #ILF76,R2       ;DONE??
1183            024736  103402                          BLO     13$             ;YES !!
1184            024740  000137  024232                  JMP     2$              ;TEST NEXT FUNCTION
1185            024744                  13$:
1186
1187
                                        ;;*****************************************************************
                                        ;*TEST 35       WRITE LOCK ERROR TEST
                                        ;;*****************************************************************
                024744
                024744  000004          TST35:  SCOPE                   ;SCOPE CALL
                024746  000240                  NOP                     ;START OF TEST
                024750  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                024754  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                024760  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                024764  012737  000035  001226  MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1188
1189            024772  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                024776  040000                  .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                025000  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                025002  000240                  NOP                     ;RETURN HERE IF ERROR
                025004  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                025006  000137  025530          JMP     13$             ;GO TO 13$ IF ERROR
1190            025012                  1$:
1191            025012  004737  042462          JSR     PC,GETSTS       ;SETUP FOR STATUS
1192            025016  012702  000000          MOV     #NOP,R2
1193            025022                  2$:
1194            025022  004737  052466          JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                025026  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
                025030  000240                  NOP                     ;RETURN HERE IF ERROR
                025032  104000                  EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                025034  000137  025530          JMP     13$             ;GO TO 13$ IF ERROR
1195            025040                  3$:
```

```
1196
1197                                    ;SET DIAGNOSTIC MODE AND DIAGNOSTIC UNIT READY, DIAGNOSTIC ON
1198                                    ;CYLINDER, AND DIAGNOSTIC WRITE LOCK
1199 025040  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
     025046  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     025054  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
     025062  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
     025066  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
     025070  000240                          NOP                     ;RETURN HERE IF ERROR
     025072  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     025074  000137  025530                  JMP     13$             ;GO TO 13$ IF ERROR
1200 025100                             4$:
1201 025100  112737  000024  001545          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
     025106  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     025114  012737  001411  001436          MOV     #DMD!MUR!MOC!MWP,RMMR10 ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC!MWP
     025122  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
     025126  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
     025130  000240                          NOP                     ;RETURN HERE IF ERROR
     025132  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     025134  000137  025530                  JMP     13$             ;GO TO 13$ IF ERROR
1202 025140                             5$:
1203
1204                                    ;SET VOLUME VALID BEFORE ISSUING FUNCTION CODE BEING TESTED
1205 025140  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
     025146  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     025154  012737  000023  001412          MOV     #PAKACK!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
     025162  004737  043016                  JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
     025166  000404                          BR      6$              ;GO TO 6$ IF NO ERROR
     025170  000240                          NOP                     ;RETURN HERE IF ERROR
     025172  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     025174  000137  025530                  JMP     13$             ;GO TO 13$ IF ERROR
1206 025200                             6$:
1207 025200  012737  000001  001412      MOV     #GO,RMCS10
1208 025206  050237  001412              BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
1209 025212  012737  104372  001416      MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
1210 025220  012737  177777  001414      MOV     #-1,RMWC0       ;DUMMY WORD COUNT
1211 025226  005037  001420              CLR     RMDA0           ;CLEAR DISK ADDRESS
1212 025232  005037  001446              CLR     RMDC0           ;CLEAR CYLINDER ADDRESS
1213 025236  012737  010000  001444      MOV     #FMT16,RMOF0    ;16 BIT FORMAT
1214 025244  012703  001545              MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
1215 025250  112723  000004              MOVB    #RMBA,(R3)+
1216 025254  112723  000002              MOVB    #RMWC,(R3)+
1217 025260  112723  000032              MOVB    #RMOF,(R3)+
1218 025264  112723  000006              MOVB    #RMDA,(R3)+
1219 025270  112723  000034              MOVB    #RMDC,(R3)+
1220 025274  112723  000000              MOVB    #RMCS1,(R3)+
1221 025300  112713  000200              MOVB    #200,(R3)
1222 025304  004737  043016              JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     025310  000404                      BR      7$              ;GO TO 7$ IF NO ERROR
     025312  000240                      NOP                     ;RETURN HERE IF ERROR
     025314  104000                      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     025316  000137  025530              JMP     13$             ;GO TO 13$ IF ERROR
1223 025322  004737  043370          7$: JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
1224
1225 025326  004737  042546              JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     025332  000404                      BR      8$              ;GO TO 8$ IF NO ERROR
     025334  000240                      NOP                     ;RETURN HERE IF ERROR
```

```
025336  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
025340  000137  025530                  JMP     13$             ;GO TO 13$ IF ERROR
1226    025344                  8$:
1227    025344  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
025350  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
025352  000240                          NOP                     ;RETURN HERE IF ERROR
025354  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
025356  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025360  000137  025530                  JMP     13$             ;GO TO 13$ IF ERROR
1228    025364  016237  070002  001140  9$:     MOV     FNCDTB(R2),$GDDAT       ;GET ENTRY FROM FUNCTION CODE TABLE
1229    025372  042737  173777  001140          BIC     #^CWLE,$GDDAT
1230    025400  013737  001352  001142          MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
1231    025406  042737  173777  001142          BIC     #^CWLE,$BDDAT
1232    025414  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS WLE STATUS CORRECT?
1233    025422  001402                          BEQ     10$             ;YES!!
1234    025424  104220                          EMT     220
1235    025426  000440                          BR      13$
1236    025430  016237  070002  001140  10$:    MOV     FNCDTB(R2),$GDDAT
1237    025436  032737  040000  001350          BIT     #ERR,RMDSI      ;WAS AN ERROR DETECTED??
1238    025444  001403                          BEQ     11$             ;NO!!
1239    025446  052737  100000  001140          BIS     #ATA,$GDDAT     ;YES - ATA SHOULD BE ON
1240    025454  042737  077777  001140  11$:    BIC     #^CATA,$GDDAT
1241    025462  013737  001350  001142          MOV     RMDSI,$BDDAT    ;GET DRIVE'S ATTENTION
1242    025470  042737  077777  001142          BIC     #^CATA,$BDDAT
1243    025476  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS ATA STATUS OK??
1244    025504  001402                          BEQ     12$             ;YES!!
1245    025506  104255                          EMT     255
1246    025510  000407                          BR      13$
1247    025512  062702  000002          12$:    ADD     #2,R2           ;GO TO NEXT FUNCTION CODE
1248    025516  022702  000076                  CMP     #ILF76,R2       ;DONE??
1249    025522  103402                          BLO     13$             ;YES !!
1250    025524  000137  025022                  JMP     2$              ;TEST NEXT FUNCTION
1251    025530                  13$:
1252
1253                            ;;**************************************************************************
                                ;*TEST 36        ERROR ABORT TESTS
                                ;;**************************************************************************
025530                  TST36:
025530  000004                  SCOPE                   ;SCOPE CALL
025532  000240                  NOP                     ;START OF TEST
025534  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
025540  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
025544  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
025550  012737  000036  001226  MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1254
1255    025556  004737  041532  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025562  054130          .WORD   054130                  ;TASK DESCRIPTOR AS FOLLOWS:
                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                        ;VERIFY PACK ACKNOWLEDGE
                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                        ;VERIFY RECALIBRATION
025564  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
025566  000240                  NOP                     ;RETURN HERE IF ERROR
025570  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
025572  000137  026062          JMP     9$              ;GO TO 9$ IF ERROR
```

```
1256 025576                                    1$:
1257
1258                                           ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
     025576  004737  042462                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
1259 025602  012702  000000                        MOV     #NOP,R2         ;R2 = FUNCTION CODE
1260 025606                                    2$:
1261 025606  004737  052466                        JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     025612  000404                              BR      3$              ;GO TO 3$ IF NO ERROR
     025614  000240                              NOP                     ;RETURN HERE IF ERROR
     025616  104000                              EMT                     ;ERROR # DEFINED BY CNTCLR SUBROUTINE
     025620  000137  026062                      JMP     9$              ;GO TO 9$ IF ERROR
1262 025624                                    3$:
1263 025624  112737  000014  001545              MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
     025632  112737  000200  001546              MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     025640  012737  040000  001426              MOV     #UNS,RMER10     ;SET RMER1 OUTPUT BUFFER = UNS
     025646  004737  043016                      JSR     PC,PUT          ;GO WRITE RMER1 VIA PUT SUBROUTINE
     025652  000404                              BR      4$              ;GO TO 4$ IF NO ERROR
     025654  000240                              NOP                     ;RETURN HERE IF ERROR
     025656  104000                              EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     025660  000137  026062                      JMP     9$              ;GO TO 9$ IF ERROR
1264 025664                                    4$:
1265 025664  012737  000001  001412              MOV     #GO,RMCS10
1266 025672  050237  001412                      BIS     R2,RMCS10       ;WRITE FUNCTION CODE IN BUFFER
1267 025676  012737  104372  001416              MOV     #BUFONE,RMBA0   ;DUMMY BUS ADDRESS
1268 025704  012737  177777  001414              MOV     #-1,RMWC0       ;DUMMY WORD COUNT
1269 025712  012737  000000  001420              MOV     #0,RMDA0        ;CLEAR DISK ADDRESS
1270 025720  012737  000000  001446              MOV     #0,RMDC0        ;CLEAR CYLINDER ADDRESS
1271 025726  012737  010000  001444              MOV     #FMT16,RMOF0    ;16 BIT FORMAT
1272 025734  012703  001545                      MOV     #PUTINX,R3      ;WRITE REGISTER INDEX TABLE
1273 025740  112723  000004                      MOVB    #RMBA,(R3)+
1274 025744  112723  000002                      MOVB    #RMWC,(R3)+
1275 025750  112723  000032                      MOVB    #RMOF,(R3)+
1276 025754  112723  000006                      MOVB    #RMDA,(R3)+
1277 025760  112723  000034                      MOVB    #RMDC,(R3)+
1278 025764  112723  000000                      MOVB    #RMCS1,(R3)+
1279 025770  112713  000200                      MOVB    #200,(R3)
1280 025774  004737  043016                      JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     026000  000402                              BR      5$              ;GO TO 5$ IF NO ERROR
     026002  000240                              NOP                     ;RETURN HERE IF ERROR
     026004  104000                              EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
1281 026006                                    5$:
1282
1283                                           ;WAIT FOR COMMAND TO COMPLETE
     026006  004737  043370                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
1284
1285 026012  004737  042546                      JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     026016  000404                              BR      6$              ;GO TO 6$ IF NO ERROR
     026020  000240                              NOP                     ;RETURN HERE IF ERROR
     026022  104000                              EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     026024  000137  026062                      JMP     9$              ;GO TO 9$ IF ERROR
1286 026030                                    6$:
1287 026030  004737  043564                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     026034  000405                              BR      7$              ;GO TO 7$ IF NO ERROR
     026036  000240                              NOP                     ;RETURN HERE IF ERROR
     026040  104000                              EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     026042  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     026044  000137  026062                      JMP     9$              ;GO TO 9$ IF ERROR
```

```
1288 026050                                7$:
1289 026050                                8$:
1290 026050  062702  000002                      ADD     #2,R2               ;ADVANCE FUNCTION CODE
1291 026054  022702  000076                      CMP     #ILF76,R2           ;DONE ALL COMMANDS
1292 026060  103252                              BHIS    2$                  ;NO !!
1293 026062                                9$:
1294
1295                                       ;;*****************************************************************
                                           ;*TEST 37       RMR TEST
                                           ;;*****************************************************************
     026062                                TST37:
     026062  000004                              SCOPE                       ;SCOPE CALL
     026064  000240                              NOP                         ;START OF TEST
     026066  012706  001100                      MOV     #STACK,SP           ;INITIALIZE STACK POINTER
     026072  013700  001276                      MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
     026076  013701  001466                      MOV     TSTQUE,R1           ;(R1) = DEVICE BEING TESTED
     026102  012737  000037  001226              MOV     #37,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
1296
1297 026110  004737  041532                      JSR     PC,TSTPRP           ;PREPARE DEVICE FOR TEST
     026114  040000                              .WORD   040000              ;TASK DESCRIPTOR AS FOLLOWS:
                                                                             ;CLEAR CONTROLLER & SELECT DEVICE
     026116  000404                              BR      1$                  ;GO TO 1$ IF NO ERROR
     026120  000240                              NOP                         ;RETURN HERE IF ERROR
     026122  104000                              EMT                         ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     026124  000137  026522                      JMP     11$                 ;GO TO 11$ IF ERROR
1298 026130                                1$:
1299
1300                                       ;SETUP GET INDEX TABLE TO READ ALL REGISTERS
     026130  004737  042462                      JSR     PC,GETSTS           ;GO TO GETSTS SUBROUTINE
1301 026134                                2$:
1302 026134  112737  000024  001545              MOVB    #RMMR1,PUTINX       ;SETUP PUT INDEX TABLE
     026142  112737  000200  001546              MOVB    #200,PUTINX+1       ;SET TERMINATOR BYTE
     026150  012737  000001  001436              MOV     #DMD,RMMR10         ;SET RMMR1 OUTPUT BUFFER = DMD
     026156  004737  043016                      JSR     PC,PUT              ;GO WRITE RMMR1 VIA PUT SUBROUTINE
     026162  000404                              BR      3$                  ;GO TO 3$ IF NO ERROR
     026164  000240                              NOP                         ;RETURN HERE IF ERROR
     026166  104000                              EMT                         ;ERROR DEFINED BY PUT SUBROUTINE
     026170  000137  026522                      JMP     11$                 ;GO TO 11$ IF ERROR
1303 026174                                3$:
1304 026174  112737  000024  001545              MOVB    #RMMR1,PUTINX       ;SETUP PUT INDEX TABLE
     026202  112737  000200  001546              MOVB    #200,PUTINX+1       ;SET TERMINATOR BYTE
     026210  012737  001401  001436              MOV     #DMD!MUR!MOC,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC
     026216  004737  043016                      JSR     PC,PUT              ;GO WRITE RMMR1 VIA PUT SUBROUTINE
     026222  000404                              BR      4$                  ;GO TO 4$ IF NO ERROR
     026224  000240                              NOP                         ;RETURN HERE IF ERROR
     026226  104000                              EMT                         ;ERROR DEFINED BY PUT SUBROUTINE
     026230  000137  026522                      JMP     11$                 ;GO TO 11$ IF ERROR
1305 026234                                4$:
1306 026234  112737  000000  001545              MOVB    #RMCS1,PUTINX       ;SETUP PUT INDEX TABLE
     026242  112737  000200  001546              MOVB    #200,PUTINX+1       ;SET TERMINATOR BYTE
     026250  012737  000023  001412              MOV     #PAKACK!GO,RMCS10       ;SET RMCS1 OUTPUT BUFFER = PAKACK!GO
     026256  004737  043016                      JSR     PC,PUT              ;GO WRITE RMCS1 VIA PUT SUBROUTINE
     026262  000404                              BR      5$                  ;GO TO 5$ IF NO ERROR
     026264  000240                              NOP                         ;RETURN HERE IF ERROR
     026266  104000                              EMT                         ;ERROR DEFINED BY PUT SUBROUTINE
     026270  000137  026522                      JMP     11$                 ;GO TO 11$ IF ERROR
1307 026274                                5$:
```

```
1308 026274 004737 042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     026300 000404                 BR      6$              ;GO TO 6$ IF NO ERROR
     026302 000240                 NOP                     ;RETURN HERE IF ERROR
     026304 104000                 EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     026306 000137 026522          JMP     11$             ;GO TO 11$ IF ERROR
1309 026312                    6$:
1310 026312 000240                 NOP
1311 026314 112737 000024 001545   MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
     026322 112737 000200 001546   MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     026330 012737 041401 001436   MOV     #DMD!MUR!MOC!DBEN,RMMR1O        ;SET RMMR1 OUTPUT BUFFER = DMD!MUR!MOC!DBEN
     026336 004737 043016          JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
     026342 000404                 BR      7$              ;GO TO 7$ IF NO ERROR
     026344 000240                 NOP                     ;RETURN HERE IF ERROR
     026346 104000                 EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     026350 000137 026522          JMP     11$             ;GO TO 11$ IF ERROR
1312 026354                    7$:
1313 026354 112737 000000 001545   MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
     026362 112737 000200 001546   MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     026370 012737 000001 001412   MOV     #NOP!GO,RMCS1O  ;SET RMCS1 OUTPUT BUFFER = NOP!GO
     026376 004737 043016          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
     026402 000404                 BR      8$              ;GO TO 8$ IF NO ERROR
     026404 000240                 NOP                     ;RETURN HERE IF ERROR
     026406 104000                 EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     026410 000137 026522          JMP     11$             ;GO TO 11$ IF ERROR
1314 026414                    8$:
1315 026414 112737 000000 001545   MOVB    #RMCS1,PUTINX   ;SETUP PUT INDEX TABLE
     026422 112737 000200 001546   MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
     026430 012737 000015 001412   MOV     #OFFSET!GO,RMCS1O       ;SET RMCS1 OUTPUT BUFFER = OFFSET!GO
     026436 004737 043016          JSR     PC,PUT          ;GO WRITE RMCS1 VIA PUT SUBROUTINE
     026442 000404                 BR      9$              ;GO TO 9$ IF NO ERROR
     026444 000240                 NOP                     ;RETURN HERE IF ERROR
     026446 104000                 EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
     026450 000137 026522          JMP     11$             ;GO TO 11$ IF ERROR
1316 026454                    9$:
1317 026454 004737 042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     026460 000404                 BR      10$             ;GO TO 10$ IF NO ERROR
     026462 000240                 NOP                     ;RETURN HERE IF ERROR
     026464 104000                 EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     026466 000137 026522          JMP     11$             ;GO TO 11$ IF ERROR
1318 026472                    10$:
1319 026472 000240                 NOP
1320 026474 032737 000004 001352   BIT     #RMR,RMER1I     ;IS RMR SET ??
1321 026502 001007                 BNE     11$             ;YES !!
1322 026504 012737 000004 001140   MOV     #RMR,$GDDAT     ;EXPECTED STATUS
1323 026512 013737 001352 001142   MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
1324 026520 104222                 EMT     222
1325 026522                    11$:
1326
1327                               ;;*****************************************************************
                                   ;*TEST 40        PARITY ERROR TEST
                                   ;;*****************************************************************
     026522                    TST40:
     026522 000004                 SCOPE                   ;SCOPE CALL
     026524 000240                 NOP                     ;START OF TEST
     026526 012706 001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     026532 013700 001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     026536 013701 001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
                026542  012737  000040  001226          MOV     #40,$TESTN      ;:SET TEST NUMBER IN APT MAIL BOX
1328
1329            026550  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                026554  040000                          .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                026556  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                026560  000240                          NOP                     ;RETURN HERE IF ERROR
                026562  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                026564  000137  026742                  JMP     7$              ;GO TO 7$ IF ERROR
1330            026570                          1$:
1331            026570  012702  000001                  MOV     #1,R2           ;R2 = DATA PATTERN
1332            026574                          2$:
1333            026574  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                026600  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                026602  000240                          NOP                     ;RETURN HERE IF ERROR
                026604  104000                          EMT                     ;ERROR # DEFINED BY CNTCLR SUBROUTINE
                026606  000137  026742                  JMP     7$              ;GO TO 7$ IF ERROR
1334            026612                          3$:
1335            026612  111103                  MOVB    (R1),R3         ;SETUP RMCS2
1336            026614  042703  177770          BIC     #^CUNTMSK,R3
1337            026620  052703  000020          BIS     #PAT,R3
1338            026624  010337  001422          MOV     R3,RMCS20       ;OUTPUT VALUE TO RMCS2
1339            026630  010237  001420          MOV     R2,RMDA0        ;VALUE TO RMDA
1340            026634  012703  001545          MOV     #PUTINX,R3      ;WRITE REGISTER OUTPUT INDEX
1341            026640  112723  000010          MOVB    #RMCS2,(R3)+
1342            026644  112723  000006          MOVB    #RMDA,(R3)+
1343            026650  112723  000200          MOVB    #200,(R3)+
1344
1345            026654  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                026660  000402                          BR      4$              ;GO TO 4$ IF NO ERROR
                026662  000240                          NOP                     ;RETURN HERE IF ERROR
                026664  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
1346            026666                          4$:
1347            026666  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                026672  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                026674  000240                          NOP                     ;RETURN HERE IF ERROR
                026676  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                026700  000137  026742                  JMP     7$              ;GO TO 7$ IF ERROR
1348            026704                          5$:
1349            026704  032737  000010  001352          BIT     #PAR,RMER1I     ;IS PARITY ERROR SET ??
1350            026712  001011                          BNE     6$              ;YES !!
1351            026714  012737  000010  001140          MOV     #PAR,$GDDAT     ;EXPECTED STATUS
1352            026722  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
1353            026730  010237  001174                  MOV     R2,$TMP0        ;DATA PATTERN
1354            026734  104223                          EMT     223
1355            026736                          6$:
1356            026736  006302                          ASL     R2              ;ADVANCE DATA PATTERN
1357            026740  001315                          BNE     2$              ;BRANCH IF NOT DONE
1358            026742                          7$:
1359
1360                                            ;;*****************************************************************
                                                ;*TEST 41        ILLEGAL REGISTER TEST
                                                ;;*****************************************************************
                026742                          TST41:
                026742  000004                          SCOPE                   ;SCOPE CALL
                026744  000240                          NOP                     ;START OF TEST
                026746  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
                026752  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                026756  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                026762  012737  000041  001226  MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
     1361
     1362       026770  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                026774  040000                  .WORD   040000          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                026776  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                027000  000240                  NOP                     ;RETURN HERE IF ERROR
                027002  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                027004  000137  027270          JMP     10$             ;GO TO 10$ IF ERROR
     1363       027010                  1$:
     1364       027010  005005                  CLR     R5              ;R5 = EXPECTED STATUS
     1365       027012  004737  042462          JSR     PC,GETSTS       ;SETUP FOR STATUS
     1366       027016  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
     1367       027022  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
     1368       027026  012737  027266  000004  MOV     #9$,ERRVEC      ;SETUP FOR BUS TIMEOUT
     1369       027034  012737  000300  000006  MOV     #PR6,ERRVEC+2
     1370       027042  005002                  CLR     R2              ;R2 = REGISTER INDEX
     1371       027044                  2$:
     1372       027044  004737  052466          JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                027050  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
                027052  000240                  NOP                     ;RETURN HERE IF ERROR
                027054  104000                  EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                027056  000137  027202          JMP     7$              ;GO TO 7$ IF ERROR
     1373       027062  010003          3$:     MOV     R0,R3           ;R3 = REGISTER ADDRESS
     1374       027064  060203                  ADD     R2,R3
     1375       027066  005013                  CLR     (R3)            ;CLEAR THE REGISTER
     1376       027070  004737  050774          JSR     PC,DEVSEL       ;GO SELECT DEVICE
                027074  000404                  BR      4$              ;GO TO 4$ IF NO ERROR
                027076  000240                  NOP                     ;RETURN HERE IF ERROR
                027100  104000                  EMT                     ;ERROR # DEFINED BY DEVSEL SUBROUTINE
                027102  000137  027270          JMP     10$             ;GO TO 10$ IF ERROR
     1377       027106                  4$:
     1378       027106  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                027112  000404                  BR      5$              ;GO TO 5$ IF NO ERROR
                027114  000240                  NOP                     ;RETURN HERE IF ERROR
                027116  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                027120  000137  027202          JMP     7$              ;GO TO 7$ IF ERROR
     1379       027124                  5$:
     1380       027124  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                027130  000405                  BR      6$              ;GO TO 6$ IF NO ERROR
                027132  000240                  NOP                     ;RETURN HERE IF ERROR
                027134  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                027136  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                027140  000137  027202          JMP     7$              ;GO TO 7$ IF ERROR
     1381       027144  010537  001140  6$:     MOV     R5,$GDDAT
     1382       027150  013737  001352  001142  MOV     RMER1I,$BDDAT   ;GET DRIVE'S ILR STATUS
     1383       027156  042737  177775  001142  BIC     #^CILR,$BDDAT
     1384       027164  023737  001140  001142  CMP     $GDDAT,$BDDAT   ;IS ILR STATUS OK??
     1385       027172  001403                  BEQ     7$              ;YES!!
     1386       027174  010237  001174          MOV     R2,$TMP0        ;SAVE R2 FOR ERROR DATA
     1387       027200  104256                  EMT     256
     1388
     1389                                       ;THE FOLLOWING CODING FOR HANDLE RH70
     1390                                       ;RH 70 REGISTER NUMBERS IS EITHER 22 OR 32
     1391       027202  062702  000002  7$:     ADD     #2,R2           ;INCREMENT THE REGISTER ADDRESS
```

```
1392 027206 022702 000050          CMP     #50,R2          ;TIME TO CHECK THE EXTEND ADDRESS REG ?
1393 027212 101314                 BHI     2$              ;BRANCH IF NOT
1394 027214 103420                 BLO     8$              ;BRANCH IF ALREADY SET UP
1395 027216 012705 000002          MOV     #ILR,R5         ;EXCEPT ILR HAPPEND
1396 027222 012760 001400 000000   MOV     #A16!A17,RMCS1(R0)      ;SET EXTENDED ADDRESS BITS
1397 027230 012702 000054          MOV     #54,R2          ;SET ADDRESS TO 54,IF 22 REGISTERS
1398 027234 016003 000050          MOV     50(R0),R3       ;IS THIS 22 REG RH70 ?
1399 027240 042703 177774          BIC     #177774,R3      ;LEFT ONLY BIT 0 AND BIT 1
1400 027244 022703 000003          CMP     #BIT0!BIT1,R3   ;BIT 0 AND BIT 1 SET AT THE SAME TIME ?
1401 027250 001402                 BEQ     8$              ;BRANCH IF SO
1402 027252 012702 000050          MOV     #50,R2          ;OTHERWISE,SET ADDRESS TO 50
1403 027256 022702 000074    8$:   CMP     #74,R2          ;ALL REGISTERS CHECKED ?
1404 027262 101402                 BLOS    10$             ;BRANCH IF SO
1405 027264 000667                 BR      2$              ;NEXT REGISTER
1406
1407 027266 022626           9$:   CMP     (SP)+,(SP)+
1408 027270                  10$:
     027270 012637 000006          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1409 027274 012637 000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1410
1411                         ;;*****************************************************************
                             ;*TEST 42        SEEK FIRST CYLINDER
                             ;;*****************************************************************
     027300                  TST42:
     027300 000004                  SCOPE                   ;SCOPE CALL
     027302 000240                  NOP                     ;START OF TEST
     027304 012706 001100           MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     027310 013700 001276           MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     027314 013701 001466           MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     027320 012737 000001 001206    MOV     #1,$TIMES       ;;DO 1 ITERATION
     027326 012737 000042 001226    MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1412
1413 027334                  1$:
1414 027334 004737 041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     027340 054130                 .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                           ;CLEAR CONTROLLER & SELECT DEVICE
                                                           ;VERIFY CONTROLLER CLEAR OPERATION
                                                           ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                           ;VERIFY PACK ACKNOWLEDGE
                                                           ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                           ;VERIFY RECALIBRATION
     027342 000404                 BR      2$              ;GO TO 2$ IF NO ERROR
     027344 000240                 NOP                     ;RETURN HERE IF ERROR
     027346 104000                 EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     027350 000137 027572          JMP     10$             ;GO TO 10$ IF ERROR
1415 027354                  2$:
1416 027354 012737 000000 001446   MOV     #0,RMDCO        ;CYLINDER = 0
1417 027362 012737 000000 001420   MOV     #0,RMDAO  3$:   ;TRACK = 0, SECTOR = 0
1418 027370 012737 000005 001412   MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
1419 027376 012702 001545          MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
1420 027402 112722 000034          MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
1421 027406 112722 000006          MOVB    #RMDA,(R2)+
1422 027412 112722 000000          MOVB    #RMCS1,(R2)+
1423 027416 112722 000200          MOVB    #200,(R2)+      ;WRITE TERMINATOR
1424 027422 004737 043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     027426 000404                 BR      4$              ;GO TO 4$ IF NO ERROR
     027430 000240                 NOP                     ;RETURN HERE IF ERROR
```

```
                027432  104000                  EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                027434  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1425            027440  004737  042462  4$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
1426            027444  004737  043370          JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
1427
1428            027450  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                027454  000404                  BR      5$              ;GO TO 5$ IF NO ERROR
                027456  000240                  NOP                     ;RETURN HERE IF ERROR
                027460  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                027462  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1429            027466          5$:
1430            027466  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                027472  000405                  BR      6$              ;GO TO 6$ IF NO ERROR
                027474  000240                  NOP                     ;RETURN HERE IF ERROR
                027476  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                027500  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                027502  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1431            027506          6$:
1432            027506  004737  051206          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
                027512  000405                  BR      7$              ;GO TO 7$ IF NO ERROR
                027514  000240                  NOP                     ;RETURN HERE IF ERROR
                027516  104000                  EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                027520  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                027522  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1433            027526          7$:
1434            027526  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                027532  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
                027534  000010                  .WORD   DPE             ;MASK FOR RMER2
                027536  000405                  BR      8$              ;GO TO 8$ IF NO ERROR
                027540  000240                  NOP                     ;RETURN HERE IF ERROR
                027542  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                027544  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                027546  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1435            027552          8$:
1436            027552  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                027556  000405                  BR      9$              ;GO TO 9$ IF NO ERROR
                027560  000240                  NOP                     ;RETURN HERE IF ERROR
                027562  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                027564  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                027566  000137  027572          JMP     10$             ;GO TO 10$ IF ERROR
1437            027572          9$:
1438
1439            027572          10$:
1440
1441                            ;;*****************************************************************
                                ;*TEST 43        SEEK LAST CYLINDER
                                ;;*****************************************************************
                027572          TST43:
                027572  000004                  SCOPE                   ;SCOPE CALL
                027574  000240                  NOP                     ;START OF TEST
                027576  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                027602  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                027606  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                027612  012737  000001  001206  MOV     #1,$TIMES       ;;DO 1 ITERATION
                027620  012737  000043  001226  MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1442
1443            027626          1$:
```

T43 SEEK LAST CYLINDER

```
1444 027626  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     027632  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                             ;CLEAR CONTROLLER & SELECT DEVICE
                                                             ;VERIFY CONTROLLER CLEAR OPERATION
                                                             ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                             ;VERIFY PACK ACKNOWLEDGE
                                                             ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                             ;VERIFY RECALIBRATION
     027634  000404                  BR      2$              ;GO TO 2$ IF NO ERROR
     027636  000240                  NOP                     ;RETURN HERE IF ERROR
     027640  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     027642  000137  030064          JMP     10$             ;GO TO 10$ IF ERROR
1445 027646                  2$:
1446 027646  012737  001060  001446 3$:     MOV     #560.,RMDCO     ;CYLINDER = 560.
1447 027654  012737  000000  001420         MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1448 027662  012737  000005  001412         MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
1449 027670  012702  001545              MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
1450 027674  112722  000034              MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
1451 027700  112722  000006              MOVB    #RMDA,(R2)+
1452 027704  112722  000000              MOVB    #RMCS1,(R2)+
1453 027710  112722  000200              MOVB    #200,(R2)+      ;WRITE TERMINATOR
1454 027714  004737  043016              JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     027720  000404                      BR      4$              ;GO TO 4$ IF NO ERROR
     027722  000240                      NOP                     ;RETURN HERE IF ERROR
     027724  104000                      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     027726  000137  030064              JMP     10$             ;GO TO 10$ IF ERROR
1455 027732  004737  042462          4$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
1456 027736  004737  043370              JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
1457
1458 027742  004737  042546              JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     027746  000404                      BR      5$              ;GO TO 5$ IF NO ERROR
     027750  000240                      NOP                     ;RETURN HERE IF ERROR
     027752  104000                      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     027754  000137  030064              JMP     10$             ;GO TO 10$ IF ERROR
1459 027760                  5$:
1460 027760  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     027764  000405                      BR      6$              ;GO TO 6$ IF NO ERROR
     027766  000240                      NOP                     ;RETURN HERE IF ERROR
     027770  104000                      EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     027772  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     027774  000137  030064              JMP     10$             ;GO TO 10$ IF ERROR
1461 030000                  6$:
1462 030000  004737  051206              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
     030004  000405                      BR      7$              ;GO TO 7$ IF NO ERROR
     030006  000240                      NOP                     ;RETURN HERE IF ERROR
     030010  104000                      EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     030012  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     030014  000137  030064              JMP     10$             ;GO TO 10$ IF ERROR
1463 030020                  7$:
1464 030020  004737  050562              JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     030024  115760                      .WORD   NDTMSK          ;MASK FOR RMER1
     030026  000010                      .WORD   DPE             ;MASK FOR RMER2
     030030  000405                      BR      8$              ;GO TO 8$ IF NO ERROR
     030032  000240                      NOP                     ;RETURN HERE IF ERROR
     030034  104000                      EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     030036  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     030040  000137  030064              JMP     10$             ;GO TO 10$ IF ERROR
```

```
                                8$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      9$              ;GO TO 9$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     10$             ;GO TO 10$ IF ERROR
                                9$:

                                10$:

                                ;;*******************************************************************
                                ;*TEST 44        SEEK PRIME CYLINDERS
                                ;;*******************************************************************
                                TST44:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #2,$TIMES       ;;DO 2 ITERATIONS
                                        MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        MOV     #1,RMDCO        ;FIRST CYLINDER = 1
                                1$:
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
                                        BR      2$              ;GO TO 2$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     9$              ;GO TO 9$ IF ERROR
                                2$:
                                        MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
                                        MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
                                        MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
                                        MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+      ;WRITE TERMINATOR
                                        JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                        BR      3$              ;GO TO 3$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     9$              ;GO TO 9$ IF ERROR
                                3$:
                                        JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
                                        JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE

                                        JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                        BR      4$              ;GO TO 4$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
```

```
                030244  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                030246  000137  030376          JMP     9$              ;GO TO 9$ IF ERROR
1490            030252                  4$:
1491            030252  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                030256  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
                030260  000240                  NOP                     ;RETURN HERE IF ERROR
                030262  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                030264  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                030266  000137  030376          JMP     9$              ;GO TO 9$ IF ERROR
1492            030272                  5$:
1493            030272  004737  051206          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
                030276  000405                  BR      6$              ;GO TO 6$ IF NO ERROR
                030300  000240                  NOP                     ;RETURN HERE IF ERROR
                030302  104000                  EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                030304  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                030306  000137  030376          JMP     9$              ;GO TO 9$ IF ERROR
1494            030312                  6$:
1495            030312  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                030316  115760                  .WORD   NDTMSK          ;MASK FOR RMER1
                030320  000010                  .WORD   DPE             ;MASK FOR RMER2
                030322  000405                  BR      7$              ;GO TO 7$ IF NO ERROR
                030324  000240                  NOP                     ;RETURN HERE IF ERROR
                030326  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                030330  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                030332  000137  030376          JMP     9$              ;GO TO 9$ IF ERROR
1496            030336                  7$:
1497            030336  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                030342  000405                  BR      8$              ;GO TO 8$ IF NO ERROR
                030344  000240                  NOP                     ;RETURN HERE IF ERROR
                030346  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                030350  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                030352  000137  030376          JMP     9$              ;GO TO 9$ IF ERROR
1498            030356                  8$:
1499            030356  006337  001446          ASL     RMDCO           ;SHIFT TO NEXT PRIME CYLINDER
1500            030362  022737  001000  001446  CMP     #512.,RMDCO     ;IS THE TEST DONE??
1501            030370  103402                  BLO     9$              ;YES!!
1502            030372  000137  030126          JMP     1$              ;GO DO NEXT CYLINDER
1503            030376                  9$:
1504
1505                                    ;;*****************************************************************
                                        ;*TEST 45       SEEK ZERO DIFFERENCE
                                        ;;*****************************************************************
                030376                  TST45:
                030376  000004                  SCOPE                   ;SCOPE CALL
                030400  000240                  NOP                     ;START OF TEST
                030402  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                030406  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                030412  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                030416  012737  000031  001206  MOV     #25.,$TIMES     ;;DO 25. ITERATIONS
                030424  012737  000045  001226  MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1506
1507            030432
1508            030432  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                030436  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
```

```
                                                        ;VERIFY PACK ACKNOWLEDGE
                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                        ;VERIFY RECALIBRATION
 030440  000404                        BR      2$      ;GO TO 2$ IF NO ERROR
 030442  000240                        NOP             ;RETURN HERE IF ERROR
 030444  104000                        EMT             ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 030446  000137  030704                JMP     10$     ;GO TO 10$ IF ERROR
1509 030452                     2$:
1510 030452  012703  000003            MOV     #3,R3           ;R3 = NUMBER OF SEEKS
1511 030456  012737  000000  001446    MOV     #0,RMDCO        ;CYLINDER = 0
1512 030464  012737  000000  001420    MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1513 030472  012737  000005  001412    MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
1514 030500  012702  001545            MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
1515 030504  112722  000006            MOVB    #RMDA,(R2)+     ;WRITE REGISTER INDEX TABLE
1516 030510  112722  000034            MOVB    #RMDC,(R2)+
1517 030514  112722  000000            MOVB    #RMCS1,(R2)+
1518 030520  112722  000200            MOVB    #200,(R2)+      ;TERMINATE TABLE
1519 030524                     3$:
1520 030524  004737  043016            JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
 030530  000404                        BR      4$              ;GO TO 4$ IF NO ERROR
 030532  000240                        NOP                     ;RETURN HERE IF ERROR
 030534  104000                        EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
 030536  000137  030704                JMP     10$             ;GO TO 10$ IF ERROR
1521 030542  004737  042462     4$:    JSR     PC,GETSTS       ;SETUP FOR READING STATUS
1522 030546  004737  043370            JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
1523
1524 030552  004737  042546            JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
 030556  000404                        BR      5$              ;GO TO 5$ IF NO ERROR
 030560  000240                        NOP                     ;RETURN HERE IF ERROR
 030562  104000                        EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
 030564  000137  030704                JMP     10$             ;GO TO 10$ IF ERROR
1525 030570                     5$:
1526 030570  004737  043564            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 030574  000405                        BR      6$              ;GO TO 6$ IF NO ERROR
 030576  000240                        NOP                     ;RETURN HERE IF ERROR
 030600  104000                        EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
 030602  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 030604  000137  030704                JMP     10$             ;GO TO 10$ IF ERROR
1527 030610                     6$:
1528 030610  004737  051206            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 030614  000405                        BR      7$              ;GO TO 7$ IF NO ERROR
 030616  000240                        NOP                     ;RETURN HERE IF ERROR
 030620  104000                        EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 030622  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 030624  000137  030704                JMP     10$             ;GO TO 10$ IF ERROR
1529 030630                     7$:
1530 030630  004737  050562            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
 030634  115760                        .WORD   NDTMSK          ;MASK FOR RMER1
 030636  000010                        .WORD   DPE             ;MASK FOR RMER2
 030640  000405                        BR      8$              ;GO TO 8$ IF NO ERROR
 030642  000240                        NOP                     ;RETURN HERE IF ERROR
 030644  104000                        EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
 030646  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 030650  000137  030704                JMP     10$             ;GO TO 10$ IF ERROR
1531 030654                     8$:
1532 030654  004737  044416            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 030660  000405                        BR      9$              ;GO TO 9$ IF NO ERROR
```

```
030662  000240                          NOP                     ;RETURN HERE IF ERROR
030664  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
030666  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
030670  000137  030704                  JMP     10$             ;GO TO 10$ IF ERROR
1533 030674                     9$:
1534 030674  005303                    DEC     R3              ;DONE ALL SEEKS??
1535 030676  001402                    BEQ     10$             ;YES!!
1536 030700  000137  030524            JMP     3$              ;NO - GO DO NEXT SEEK
1537 030704                     10$:
1538
1539                            ;;*******************************************************************
                                ;*TEST 46        SEEK MAXIMUM DIFFERENCE FORWARD
                                ;;*******************************************************************
030704                          TST46:
030704  000004                          SCOPE                   ;SCOPE CALL
030706  000240                          NOP                     ;START OF TEST
030710  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
030714  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
030720  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
030724  012737  000031  001206          MOV     #25.,$TIMES     ;;DO 25. ITERATIONS
030732  012737  000046  001226          MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1540
1541 030740                     1$:
1542 030740  004737  041532            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
030744  052130                          .WORD   052130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE DRIVE
                                                                ;VERIFY RECALIBRATION
030746  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
030750  000240                          NOP                     ;RETURN HERE IF ERROR
030752  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
030754  000137  031172                  JMP     8$              ;GO TO 8$ IF ERROR
1543 030760                     2$:
1544 030760  012737  001060  001446      MOV     #560.,RMDCO     ;LAST CYLINDER
1545 030766  012737  000000  001420      MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1546 030774  012737  000005  001412      MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND IN BUFFER
1547 031002  012702  001545              MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
1548 031006  112722  000034              MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
1549 031012  112722  000006              MOVB    #RMDA,(R2)+
1550 031016  112722  000000              MOVB    #RMCS1,(R2)+
1551 031022  112722  000200              MOVB    #200,(R2)+      ;WRITE TERMINATOR
1552 031026  004737  043016              JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
031032  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
031034  000240                          NOP                     ;RETURN HERE IF ERROR
031036  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
031040  000137  031172                  JMP     8$              ;GO TO 8$ IF ERROR
1553 031044  004737  042462      3$:     JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
1554 031050  004737  043370              JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
1555
1556 031054  004737  042546              JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
031060  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
031062  000240                          NOP                     ;RETURN HERE IF ERROR
031064  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
031066  000137  031172                  JMP     8$              ;GO TO 8$ IF ERROR
```

```
1557 031072                                4$:
1558 031072  004737  043564                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     031076  000405                              BR      5$              ;GO TO 5$ IF NO ERROR
     031100  000240                              NOP                     ;RETURN HERE IF ERROR
     031102  104000                              EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     031104  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     031106  000137  031172                      JMP     8$              ;GO TO 8$ IF ERROR
1559 031112                                5$:
1560 031112  004737  051206                      JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
     031116  000405                              BR      6$              ;GO TO 6$ IF NO ERROR
     031120  000240                              NOP                     ;RETURN HERE IF ERROR
     031122  104000                              EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     031124  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     031126  000137  031172                      JMP     8$              ;GO TO 8$ IF ERROR
1561 031132                                6$:
1562 031132  004737  050562                      JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     031136  115760                              .WORD   NDTMSK          ;MASK FOR RMER1
     031140  000010                              .WORD   DPE             ;MASK FOR RMER2
     031142  000405                              BR      7$              ;GO TO 7$ IF NO ERROR
     031144  000240                              NOP                     ;RETURN HERE IF ERROR
     031146  104000                              EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     031150  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     031152  000137  031172                      JMP     8$              ;GO TO 8$ IF ERROR
1563 031156                                7$:
1564 031156  004737  044416                      JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     031162  000403                              BR      8$              ;GO TO 8$ IF NO ERROR
     031164  000240                              NOP                     ;RETURN HERE IF ERROR
     031166  104000                              EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     031170  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
1565 031172                                8$:
1566
1567                                       ;;*******************************************************************
                                           ;*TEST 47        SEEK ADJACENT FORWARD
                                           ;;*******************************************************************
     031172                                TST47:
     031172  000004                              SCOPE                   ;SCOPE CALL
     031174  000240                              NOP                     ;START OF TEST
     031176  012706  001100                      MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     031202  013700  001276                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     031206  013701  001466                      MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     031212  012737  000047  001226              MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1568
1569 031220                                1$:
1570 031220  004737  041532                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     031224  054130                              .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                         ;CLEAR CONTROLLER & SELECT DEVICE !
                                                                         ;VERIFY CONTROLLER CLEAR OPERATION,
                                                                         ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                         ;VERIFY PACK ACKNOWLEDGE
                                                                         ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                         ;VERIFY RECALIBRATION
     031226  000404                              BR      2$              ;GO TO 2$ IF NO ERROR
     031230  000240                              NOP                     ;RETURN HERE IF ERROR
     031232  104000                              EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     031234  000137  031474                      JMP     10$             ;GO TO 10$ IF ERROR
1571 031240                                2$:
1572 031240  012737  000000  001446              MOV     #0,RMDCO        ;CYLINDER = 0
```

```
1573 031246 012737 000000 001420          MOV     #0,RMDA0          ;TRACK = 0, SECTOR = 0
1574 031254 012737 000005 001412          MOV     #SEEK!GO,RMCS10   ;FUNCTION CODE FOR SEEK
1575 031262 012702 001545                 MOV     #PUTINX,R2        ;R2 POINTS TO REGISTER INDEX
1576 031266 112722 000034                 MOVB    #RMDC,(R2)+       ;WRITE REGISTER INDEX TABLE
1577 031272 112722 000006                 MOVB    #RMDA,(R2)+
1578 031276 112722 000000                 MOVB    #RMCS1,(R2)+
1579 031302 112722 000200                 MOVB    #200,(R2)+        ;TERMINATE REGISTER TABLE
1580 031306                         3$:
1581 031306 004737 043016                 JSR     PC,PUT            ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     031312 000404                        BR      4$                ;GO TO 4$ IF NO ERROR
     031314 000240                        NOP                       ;RETURN HERE IF ERROR
     031316 104000                        EMT                       ;ERROR # DEFINED BY PUT SUBROUTINE
     031320 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1582 031324 004737 042462           4$:   JSR     PC,GETSTS         ;SETUP FOR STATUS FETCH
1583 031330 004737 043370                 JSR     PC,TIMOUT         ;WAIT FOR COMPLETION
1584
1585 031334 004737 042546                 JSR     PC,GET            ;GO READ REGISTER(S) WITH GET SUBROUTINE
     031340 000404                        BR      5$                ;GO TO 5$ IF NO ERROR
     031342 000240                        NOP                       ;RETURN HERE IF ERROR
     031344 104000                        EMT                       ;ERROR # DEFINED BY GET SUBROUTINE
     031346 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1586 031352                         5$:
1587 031352 004737 043564                 JSR     PC,PRIERR         ;GO CHECK FOR PRIMARY ERRORS
     031356 000405                        BR      6$                ;GO TO 6$ IF NO ERROR
     031360 000240                        NOP                       ;RETURN HERE IF ERROR
     031362 104000                        EMT                       ;ERROR # DEFINED BY PRIERR SUBROUTINE
     031364 004736                        JSR     PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     031366 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1588 031372                         6$:
1589 031372 004737 051206                 JSR     PC,SEKSTS         ;GO VERIFY RESULTS OF SEEK OPERATION
     031376 000405                        BR      7$                ;GO TO 7$ IF NO ERROR
     031400 000240                        NOP                       ;RETURN HERE IF ERROR
     031402 104000                        EMT                       ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     031404 004736                        JSR     PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     031406 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1590 031412                         7$:
1591 031412 004737 050562                 JSR     PC,CMPERRSTS      ;CHECK ANY ERRORS NOT MASKED
     031416 115760                        .WORD   NDTMSK            ;MASK FOR RMER1
     031420 000010                        .WORD   DPE               ;MASK FOR RMER2
     031422 000405                        BR      8$                ;GO TO 8$ IF NO ERROR
     031424 000240                        NOP                       ;RETURN HERE IF ERROR
     031426 104000                        EMT                       ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     031430 004736                        JSR     PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     031432 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1592 031436                         8$:
1593 031436 004737 044416                 JSR     PC,SECERR         ;GO CHECK FOR SECONDARY ERRORS
     031442 000405                        BR      9$                ;GO TO 9$ IF NO ERROR
     031444 000240                        NOP                       ;RETURN HERE IF ERROR
     031446 104000                        EMT                       ;ERROR # DEFINED BY SECERR SUBROUTINE
     031450 004736                        JSR     PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     031452 000137 031474                 JMP     10$               ;GO TO 10$ IF ERROR
1594 031456                         9$:
1595 031456 005737 001446                 TST     RMDC0             ;FIRST TIME THROUGH??
1596 031462 001004                        BNE     10$               ;NO!!
1597 031464 005237 001446                 INC     RMDC0             ;YES - SEEK TO ADJACENT CYLINDER FORWARD
1598 031470 000137 031306                 JMP     3$
1599 031474                         10$:
```

```
1600
1601                                    ;;*****************************************************************
                                        ;*TEST 50        SEEK ADJACENT REVERSE
                                        ;;*****************************************************************
                                        TST50:
        031474
        031474  000004                          SCOPE                   ;SCOPE CALL
        031476  000240                          NOP                     ;START OF TEST
        031500  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
        031504  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        031510  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
        031514  012737  000050  001226          MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1602
1603    031522                          1$:
1604    031522  004737  041532                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
        031526  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
        031530  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
        031532  000240                          NOP                     ;RETURN HERE IF ERROR
        031534  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
        031536  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
1605    031542                          2$:
1606    031542  012737  000001  001446          MOV     #1,RMDC0        ;CYLINDER = 1
1607    031550  012737  000000  001420          MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
1608    031556  012737  000005  001412          MOV     #SEEK!GO,RMCS10 ;FUNCTION CODE FOR SEEK
1609    031564  012702  001545                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER INDEX
1610    031570  112722  000006                  MOVB    #RMDA,(R2)+     ;WRITE REGISTER INDEX TABLE
1611    031574  112722  000034                  MOVB    #RMDC,(R2)+
1612    031600  112722  000000                  MOVB    #RMCS1,(R2)+
1613    031604  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
1614    031610                          3$:
1615    031610  004737  043016                 JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
        031614  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
        031616  000240                          NOP                     ;RETURN HERE IF ERROR
        031620  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
        031622  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
1616    031626  004737  042462          4$:     JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
1617    031632  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
1618
1619    031636  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
        031642  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
        031644  000240                          NOP                     ;RETURN HERE IF ERROR
        031646  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
        031650  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
1620    031654                          5$:
1621    031654  004737  043564                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
        031660  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
        031662  000240                          NOP                     ;RETURN HERE IF ERROR
        031664  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
        031666  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
        031670  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
1622    031674                          6$:
1623    031674  004737  051206                 JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
```

```
                031700  000405                          BR      7$              ;GO TO 7$ IF NO ERROR
                031702  000240                          NOP                     ;RETURN HERE IF ERROR
                031704  104000                          EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                031706  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                031710  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
     1624       031714                          7$:
     1625       031714  004737  050562                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                031720  115760                          .WORD   NDTMSK          ;MASK FOR RMER1
                031722  000010                          .WORD   DPE             ;MASK FOR RMER2
                031724  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
                031726  000240                          NOP                     ;RETURN HERE IF ERROR
                031730  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                031732  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                031734  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
     1626       031740                          8$:
     1627       031740  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                031744  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
                031746  000240                          NOP                     ;RETURN HERE IF ERROR
                031750  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                031752  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                031754  000137  032000                  JMP     10$             ;GO TO 10$ IF ERROR
     1628       031760                          9$:
     1629       031760  022737  000001  001446          CMP     #1,RMDCO        ;IS THIS THE FIRST SEEK??
     1630       031766  001004                          BNE     10$             ;NO!!
     1631       031770  005337  001446                  DEC     RMDCO           ;YES - SEEK TO ADJACENT CYLINDER
     1632       031774  000137  031610                  JMP     3$              ;GO SEEK
     1633       032000                          10$:
     1634
     1635                                        ;;*******************************************************************
                                                 ;*TEST 51        SEEK INVALID SECTOR
                                                 ;;*******************************************************************
                032000                          TST51:
                032000  000004                          SCOPE                   ;SCOPE CALL
                032002  000240                          NOP                     ;START OF TEST
                032004  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                032010  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                032014  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                032020  012737  000051  001226          MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
     1636
     1637       032026                          1$:
     1638       032026  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                032032  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                032034  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                032036  000240                          NOP                     ;RETURN HERE IF ERROR
                032040  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                032042  000137  032416                  JMP     14$             ;GO TO 14$ IF ERROR
     1639       032046                          2$:
     1640       032046  012737  000005  001412          MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
     1641       032054  012737  000000  001444          MOV     #0,RMOFO        ;FORMAT 18 BIT
     1642       032062  012737  000000  001446          MOV     #0,RMDCO        ;CYLINDER = 0
     1643       032070  012737  000036  001420          MOV     #30.,RMDAO      ;INVALID SECTOR = 30., TRACK = 0
```

```
1644 032076  012702  001545          MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
1645 032102  112722  000032          MOVB    #RMOF,(R2)+     ;WRITE REGISTER INDEX TABLE
1646 032106  112722  000006          MOVB    #RMDA,(R2)+
1647 032112  112722  000034          MOVB    #RMDC,(R2)+
1648 032116  112722  000000          MOVB    #RMCS1,(R2)+
1649 032122  112722  000200          MOVB    #200,(R2)+      ;TERMINATE TABLE
1650 032126  004737  042462          JSR     PC,GETSTS       ;SETUP INPUT REGISTER INDEX
1651 032132                      3$:
1652 032132  004737  052466          JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     032136  000404                  BR      4$              ;GO TO 4$ IF NO ERROR
     032140  000240                  NOP                     ;RETURN HERE IF ERROR
     032142  104000                  EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     032144  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1653 032150                      4$:
1654 032150  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     032154  000404                  BR      5$              ;GO TO 5$ IF NO ERROR
     032156  000240                  NOP                     ;RETURN HERE IF ERROR
     032160  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     032162  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1655 032166                      5$:
1656 032166  004737  052604          JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
     032172  000405                  BR      6$              ;GO TO 6$ IF NO ERROR
     032174  000240                  NOP                     ;RETURN HERE IF ERROR
     032176  104000                  EMT                     ;ERROR # DEFINED BY CLRSTS SUBROUTINE
     032200  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032202  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1657 032206                      6$:
1658 032206  004737  043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     032212  000404                  BR      7$              ;GO TO 7$ IF NO ERROR
     032214  000240                  NOP                     ;RETURN HERE IF ERROR
     032216  104000                  EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     032220  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1659 032224                      7$:
1660 032224  004737  043370          JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
1661
1662 032230  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     032234  000404                  BR      8$              ;GO TO 8$ IF NO ERROR
     032236  000240                  NOP                     ;RETURN HERE IF ERROR
     032240  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     032242  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1663 032246                      8$:
1664 032246  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     032252  000405                  BR      9$              ;GO TO 9$ IF NO ERROR
     032254  000240                  NOP                     ;RETURN HERE IF ERROR
     032256  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     032260  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032262  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1665 032266                      9$:
1666 032266  004737  051206          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
     032272  000405                  BR      10$             ;GO TO 10$ IF NO ERROR
     032274  000240                  NOP                     ;RETURN HERE IF ERROR
     032276  104000                  EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     032300  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032302  000137  032416          JMP     14$             ;GO TO 14$ IF ERROR
1667 032306                     10$:
1668 032306                     11$:
1669 032306  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
```

```
T51     SEEK INVALID SECTOR

032312  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
032314  000010                          .WORD   DPE                     ;MASK FOR RMER2
032316  000405                          BR      12$                     ;GO TO 12$ IF NO ERROR
032320  000240                          NOP                             ;RETURN HERE IF ERROR
032322  104000                          EMT                             ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
032324  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
032326  000137  032416                  JMP     14$                     ;GO TO 14$ IF ERROR
1670 032332                     12$:
1671 032332  004737  044416            JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
032336  000405                          BR      13$                     ;GO TO 13$ IF NO ERROR
032340  000240                          NOP                             ;RETURN HERE IF ERROR
032342  104000                          EMT                             ;ERROR # DEFINED BY SECERR SUBROUTINE
032344  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
032346  000137  032416                  JMP     1 $                     ;GO TO 14$ IF ERROR
1672 032352                     13$:
1673 032352  005237  001420            INC     RMDA0                   ;INCREMENT SECTOR ADDRESS
1674 032356  023727  001420  000100    CMP     RMDA0,#64.              ;DONE YET ?
1675 032364  101662                    BLOS    3$                      ;NO - TEST NEXT SECTOR
1676
1677 032366  032737  010000  001444    BIT     #FMT16,RMOF0            ;WAS TEST DONE FOR 16 BIT MODE YET ?
1678 032374  001010                    BNE     14$                     ;BR IF YES
1679 032376  012737  010000  001444    MOV     #FMT16,RMOF0            ;SET 16 BIT FORMAT MODE
1680 032404  012737  000037  001420    MOV     #31.,RMDA0              ;SET INVALID SECTOR 31.
1681 032412  000137  032132            JMP     3$                      ;NO - TEST NEXT SECTOR
1682 032416                     14$:
1683
1684                            ;;*******************************************************************
                                ;*TEST 52        SEEK INVALID TRACK
                                ;;*******************************************************************
032416                          TST52:
032416  000004                          SCOPE                           ;SCOPE CALL
032420  000240                          NOP                             ;START OF TEST
032422  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
032426  013700  001276                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
032432  013701  001466                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
032436  012737  000052  001226          MOV     #52,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
1685
1686 032444  004737  041532            JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
032450  054130                          .WORD   054130                  ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
032452  000404                          BR      1$                      ;GO TO 1$ IF NO ERROR
032454  000240                          NOP                             ;RETURN HERE IF ERROR
032456  104000                          EMT                             ;ERROR # DEFINED BY TSTPRP SUBROUTINE
032460  000137  033014                  JMP     13$                     ;GO TO 13$ IF ERROR
1687 032464                     1$:
1688 032464  012737  000005  001412    MOV     #SEEK!GO,RMCS10         ;SEEK COMMAND
1689 032472  012737  000000  001446    MOV     #0,RMDC0                ;CYLINDER = 0
1690 032500  013737  001334  001420    MOV     LSTRK,RMDA0             ;LAST TRACK, SECTOR = 0
1691 032506  105237  001421            INCB    RMDA0+1                 ;SETUP FIRST INVALID TRACK
1692 032512  012737  010000  001444    MOV     #FMT16,RMOF0            ;SET 16 BIT FORMAT
1693 032520  012702  001545            MOV     #PUTINX,R2              ;R2 POINTS TO REGISTER TABLE
1694 032524  112722  000032            MOVB    #RMOF,(R2)+             ;WRITE REGISTER INDEX
```

```
1695 032530 112722 000034                MOVB    #RMDC,(R2)+     ;TABLE FOR OUTPUT
1696 032534 112722 000006                MOVB    #RMDA,(R2)+
1697 032540 112722 000000                MOVB    #RMCS1,(R2)+
1698 032544 112722 000200                MOVB    #200,(R2)+      ;TERMINATE TABLE
1699 032550 004737 042462                JSR     PC,GETSTS       ;SETUP INPUT TABLE FOR STATUS
1700 032554                          2$:
1701 032554 004737 052466                JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     032560 000404                       BR      3$              ;GO TO 3$ IF NO ERROR
     032562 000240                       NOP                     ;RETURN HERE IF ERROR
     032564 104000                       EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     032566 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1702 032572                          3$:
1703 032572 004737 042546                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     032576 000404                       BR      4$              ;GO TO 4$ IF NO ERROR
     032600 000240                       NOP                     ;RETURN HERE IF ERROR
     032602 104000                       EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     032604 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1704 032610                          4$:
1705 032610 004737 052604                JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
     032614 000405                       BR      5$              ;GO TO 5$ IF NO ERROR
     032616 000240                       NOP                     ;RETURN HERE IF ERROR
     032620 104000                       EMT                     ;ERROR # DEFINED BY CLRSTS SUBROUTINE
     032622 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032624 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1706 032630                          5$:
1707 032630 004737 043016                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     032634 000404                       BR      6$              ;GO TO 6$ IF NO ERROR
     032636 000240                       NOP                     ;RETURN HERE IF ERROR
     032640 104000                       EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     032642 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1708 032646                          6$:
1709 032646 004737 043370                JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
1710
1711 032652 004737 042546                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     032656 000404                       BR      7$              ;GO TO 7$ IF NO ERROR
     032660 000240                       NOP                     ;RETURN HERE IF ERROR
     032662 104000                       EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     032664 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1712 032670                          7$:
1713 032670 004737 043564                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     032674 000405                       BR      8$              ;GO TO 8$ IF NO ERROR
     032676 000240                       NOP                     ;RETURN HERE IF ERROR
     032700 104000                       EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     032702 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032704 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1714 032710                          8$:
1715 032710 004737 051206                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
     032714 000405                       BR      9$              ;GO TO 9$ IF NO ERROR
     032716 000240                       NOP                     ;RETURN HERE IF ERROR
     032720 104000                       EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     032722 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     032724 000137 033014                JMP     13$             ;GO TO 13$ IF ERROR
1716 032730                          9$:
1717 032730                          10$:
1718 032730 004737 050562                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     032734 117760                       .WORD   IAE!NDTMSK              ;MASK FOR RMER1
     032736 000010                       .WORD   DPE             ;MASK FOR RMER2
```

```
                032740  000405                  BR      11$             ;GO TO 11$ IF NO ERROR
                032742  000240                  NOP                     ;RETURN HERE IF ERROR
                032744  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                032746  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                032750  000137  033014          JMP     13$             ;GO TO 13$ IF ERROR
  1719          032754                  11$:
  1720          032754  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                032760  000405                  BR      12$             ;GO TO 12$ IF NO ERROR
                032762  000240                  NOP                     ;RETURN HERE IF ERROR
                032764  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                032766  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                032770  000137  033014          JMP     13$             ;GO TO 13$ IF ERROR
  1721          032774                  12$:
  1722          032774  105237  001421          INCB    RMDA0+1         ;INCREMENT TRACK ADDRESS
  1723          033000  123727  001421  000200  CMPB    RMDA0+1,#128.   ;DONE??
  1724          033006  101002                  BHI     13$             ;BR IF YES
  1725          033010  000137  032554          JMP     2$              ;TEST NEXT INVALID TRACK
  1726          033014                  13$:
  1727
  1728                                  ;;*****************************************************************
                                        ;*TEST 53        SEEK INVALID CYLINDER
                                        ;;*****************************************************************
                033014                  TST53:
                033014  000004                  SCOPE                   ;SCOPE CALL
                033016  000240                  NOP                     ;START OF TEST
                033020  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                033024  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                033030  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                033034  012737  000053  001226  MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  1729
  1730          033042  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                033046  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
                033050  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                033052  000240                  NOP                     ;RETURN HERE IF ERROR
                033054  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                033056  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
  1731          033062                  1$:
  1732          033062  012737  000005  001412  MOV     #SEEK!GO,RMCS10 ;SEEK FUNCTION CODE
  1733          033070  012737  001061  001446  MOV     #561.,RMDC0     ;START AT FIRST INVALID CYLINDER
  1734          033076  012737  000000  001420  MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
  1735          033104  012737  010000  001444  MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
  1736          033112  012702  001545          MOV     #PUTINX,R2      ;R2 POINTS TO INDEX TABLE
  1737          033116  112722  000006          MOVB    #RMDA,(R2)+     ;WRITE REGISTER TABLE
  1738          033122  112722  000034          MOVB    #RMDC,(R2)+
  1739          033126  112722  000032          MOVB    #RMOF,(R2)+
  1740          033132  112722  000000          MOVB    #RMCS1,(R2)+
  1741          033136  112722  000200          MOVB    #200,(R2)+      ;TERMINATE TABLE
  1742          033142  004737  042462          JSR     PC,GETSTS       ;SETUP FOR STATUS
  1743          033146                  2$:
  1744          033146  004737  052466          JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                033152  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
```

```
                033154  000240                  NOP                     ;RETURN HERE IF ERROR
                033156  104000                  EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                033160  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1745            033164                  3$:
1746            033164  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                033170  000404                  BR      4$              ;GO TO 4$ IF NO ERROR
                033172  000240                  NOP                     ;RETURN HERE IF ERROR
                033174  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                033176  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1747            033202                  4$:
1748            033202  004737  052604          JSR     PC,CLRSTS       ;GO VERIFY CONTROLLER CLEAR OPERATION
                033206  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
                033210  000240                  NOP                     ;RETURN HERE IF ERROR
                033212  104000                  EMT                     ;ERROR # DEFINED BY CLRSTS SUBROUTINE
                033214  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                033216  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1749            033222                  5$:
1750            033222  004737  043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                033226  000404                  BR      6$              ;GO TO 6$ IF NO ERROR
                033230  000240                  NOP                     ;RETURN HERE IF ERROR
                033232  104000                  EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                033234  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1751            033240                  6$:
1752            033240  004737  043370          JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
1753
1754            033244  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                033250  000404                  BR      7$              ;GO TO 7$ IF NO ERROR
                033252  000240                  NOP                     ;RETURN HERE IF ERROR
                033254  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                033256  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1755            033262                  7$:
1756            033262  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                033266  000405                  BR      8$              ;GO TO 8$ IF NO ERROR
                033270  000240                  NOP                     ;RETURN HERE IF ERROR
                033272  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                033274  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                033276  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1757            033302                  8$:
1758            033302  004737  051206          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
                033306  000405                  BR      9$              ;GO TO 9$ IF NO ERROR
                033310  000240                  NOP                     ;RETURN HERE IF ERROR
                033312  104000                  EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                033314  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                033316  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1759            033322                  9$:
1760            033322  004737  050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                033326  117760                  .WORD   IAE!NDTMSK              ;MASK FOR RMER1
                033330  000010                  .WORD   DPE             ;MASK FOR RMER2
                033332  000405                  BR      10$             ;GO TO 10$ IF NO ERROR
                033334  000240                  NOP                     ;RETURN HERE IF ERROR
                033336  104000                  EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                033340  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                033342  000137  033406          JMP     12$             ;GO TO 12$ IF ERROR
1761            033346                  10$:
1762            033346  004737  044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                033352  000405                  BR      11$             ;GO TO 11$ IF NO ERROR
                033354  000240                  NOP                     ;RETURN HERE IF ERROR
```

```
033356  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
033360  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033362  000137  033406                  JMP     12$             ;GO TO 12$ IF ERROR
1763 033366                     11$:
1764 033366  005237  001446            INC     RMDCO           ;INCREMENT CYLINDER ADDRESS
1765 033372  023727  001446  002000    CMP     RMDCO,#1024.    ;DONE ALL CYLINDERS ?
1766 033400  103002                    BHIS    12$             ;BR IF YES
1767 033402  000137  033146            JMP     2$              ;TEST NEXT INVALID CYLINDER
1768 033406                     12$:
1769
1770                            ;;*******************************************************************
                                ;*TEST 54        IVC SEEK TEST
                                ;;*******************************************************************
033406                          TST54:
033406  000004                          SCOPE                   ;SCOPE CALL
033410  000240                          NOP                     ;START OF TEST
033412  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
033416  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
033422  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
033426  012737  000054  001226          MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1771
1772 033434  004737  052466            JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
033440  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
033442  000240                          NOP                     ;RETURN HERE IF ERROR
033444  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
033446  000137  033746                  JMP     8$              ;GO TO 8$ IF ERROR
1773 033452                     1$:
1774 033452  112737  000024  001545    MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
033460  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
033466  012737  000001  001436          MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
033474  004737  043016                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
033500  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
033502  000240                          NOP                     ;RETURN HERE IF ERROR
033504  104000                          EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
033506  000137  033746                  JMP     8$              ;GO TO 8$ IF ERROR
1775 033512                     2$:
1776 033512  012737  000005  001412    MOV     #SEEK!GO,RMCS10
1777 033520  012737  000000  001436    MOV     #0,RMMR10
1778 033526  012737  000000  001446    MOV     #0,RMDCO        ;CYLINDER = 0
1779 033534  012737  000000  001420    MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1780 033542  012702  001545            MOV     #PUTINX,R2      ;WRITE REGISTER INDEX
1781 033546  112722  000024            MOVB    #RMMR1,(R2)+    ;TABLE
1782 033552  112722  000006            MOVB    #RMDA,(R2)+
1783 033556  112722  000034            MOVB    #RMDC,(R2)+
1784 033562  112722  000000            MOVB    #RMCS1,(R2)+
1785 033566  112722  000200            MOVB    #200,(R2)+
1786 033572  004737  042462            JSR     PC,GETSTS       ;SETUP FOR STATUS
1787 033576  004737  043016            JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
033602  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
033604  000240                          NOP                     ;RETURN HERE IF ERROR
033606  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
033610  000137  033746                  JMP     8$              ;GO TO 8$ IF ERROR
1788 033614                     3$:
1789 033614  004737  042546            JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
033620  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
033622  000240                          NOP                     ;RETURN HERE IF ERROR
033624  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
```

```
033626  000137  033746                 4$:     JMP     8$              ;GO TO 8$ IF ERROR
1790 033632
1791 033632  004737  043564                     JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     033636  000405                             BR      5$              ;GO TO 5$ IF NO ERROR
     033640  000240                             NOP                     ;RETURN HERE IF ERROR
     033642  104000                             EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     033644  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     033646  000137  033746                     JMP     8$              ;GO TO 8$ IF ERROR
1792 033652  032737  010000  001400     5$:     BIT     #IVC,RMER2I     ;DID INVALID COMMAND SET??
1793 033660  001012                             BNE     6$              ;YES!!
1794 033662  012737  010000  001140             MOV     #IVC,$GDDAT     ;GOOD DATA FOR TYPEOUT
1795 033670  013737  001400  001142             MOV     RMER2I,$BDDAT   ;BAD DATA FOR TYPEOUT
1796 033676  042737  167777  001142             BIC     #^CIVC,$BDDAT
1797 033704  104064                             EMT     64
1798 033706                             6$:
1799 033706  004737  050562                     JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     033712  115760                             .WORD   NDTMSK          ;MASK FOR RMER1
     033714  010010                             .WORD   IVC!DPE         ;MASK FOR RMER2
     033716  000405                             BR      7$              ;GO TO 7$ IF NO ERROR
     033720  000240                             NOP                     ;RETURN HERE IF ERROR
     033722  104000                             EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     033724  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     033726  000137  033746                     JMP     8$              ;GO TO 8$ IF ERROR
1800 033732                             7$:
1801 033732  004737  044416                     JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     033736  000403                             BR      8$              ;GO TO 8$ IF NO ERROR
     033740  000240                             NOP                     ;RETURN HERE IF ERROR
     033742  104000                             EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     033744  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
1802 033746                             8$:
1803
1804                                    ;;*******************************************************************
                                        ;*TEST 55        ABORT SEEK TEST
                                        ;;*******************************************************************
     033746
     033746  000004                     TST55:  SCOPE                   ;SCOPE CALL
     033750  000240                             NOP                     ;START OF TEST
     033752  012706  001100                     MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     033756  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     033762  013701  001466                     MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     033766  012737  000055  001226             MOV     #55,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1805
1806 033774  004737  041532                     JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     034000  054130                             .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
     034002  000404                             BR      1$              ;GO TO 1$ IF NO ERROR
     034004  000240                             NOP                     ;RETURN HERE IF ERROR
     034006  104000                             EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     034010  000137  034310                     JMP     9$              ;GO TO 9$ IF ERROR
1807 034014                             1$:
1808 034014  012737  040000  001426             MOV     #UNS,RMER10     ;SET UNSAFE ERROR
1809 034022  012737  000000  001446             MOV     #0,RMDC0        ;CYLINDER = 0
```

```
1810 034030 012737 000000 001420        MOV   #0,RMDA0          ;TRACK = 0, SECTOR = 0
1811 034036 012737 000005 001412        MOV   #SEEK!GO,RMCS10   ;SEEK COMMAND
1812 034044 012702 001545               MOV   #PUTINX,R2        ;SETUP REGISTER INDEX TABLE
1813 034050 112722 000006              MOVB   #RMDA,(R2)+       ;AND OUTPUT BUFFER
1814 034054 112722 000034              MOVB   #RMDC,(R2)+
1815 034060 112722 000014              MOVB   #RMER1,(R2)+
1816 034064 112722 000000              MOVB   #RMCS1,(R2)+
1817 034070 112722 000200              MOVB   #200,(R2)+
1818 034074 004737 042462               JSR   PC,GETSTS         ;SETUP FOR STATUS FETCH
1819 034100 004737 043016               JSR   PC,PUT            ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     034104 000404                      BR    2$                ;GO TO 2$ IF NO ERROR
     034106 000240                      NOP                     ;RETURN HERE IF ERROR
     034110 104000                      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     034112 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1820 034116                     2$:
1821 034116 004737 042546               JSR   PC,GET            ;GO READ REGISTER(S) WITH GET SUBROUTINE
     034122 000404                      BR    3$                ;GO TO 3$ IF NO ERROR
     034124 000240                      NOP                     ;RETURN HERE IF ERROR
     034126 104000                      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     034130 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1822 034134 032737 020000 001350 3$:    BIT   #PIP,RMDSI        ;DID DRIVE START SEEK??
1823 034142 001411                      BEQ   4$                ;NO!!
1824 034144 013737 001350 001142        MOV   RMDSI,$BDDAT      ;BAD DATA FOR TYPEOUT
1825 034152 042737 157777 001142        BIC   #^CPIP,$BDDAT
1826 034160 005037 001140               CLR   $GDDAT            ;GOOD DATA FOR TYPEOUT
1827 034164 104065                      EMT   65
1828 034166 004737 043370       4$:     JSR   PC,TIMOUT         ;WAIT FOR GO TO RESET
1829
1830 034172 004737 042546               JSR   PC,GET            ;GO READ REGISTER(S) WITH GET SUBROUTINE
     034176 000404                      BR    5$                ;GO TO 5$ IF NO ERROR
     034200 000240                      NOP                     ;RETURN HERE IF ERROR
     034202 104000                      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     034204 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1831 034210                     5$:
1832 034210 004737 043564               JSR   PC,PRIERR         ;GO CHECK FOR PRIMARY ERRORS
     034214 000405                      BR    6$                ;GO TO 6$ IF NO ERROR
     034216 000240                      NOP                     ;RETURN HERE IF ERROR
     034220 104000                      EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     034222 004736                      JSR   PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     034224 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1833 034230                     6$:
1834 034230 004737 060210               JSR   PC,STCDRVSTS      ;GO CHECK FOR CHANGES IN DRIVE STATUS
     034234 000405                      BR    7$                ;GO TO 7$ IF NO ERROR
     034236 000240                      NOP                     ;RETURN HERE IF ERROR
     034240 104000                      EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
     034242 004736                      JSR   PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     034244 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1835 034250                     7$:
1836 034250 004737 050562               JSR   PC,CMPERRSTS      ;CHECK ANY ERRORS NOT MASKED
     034254 155760                      .WORD NDTMSK!UNS                ;MASK FOR RMER1
     034256 000010                      .WORD DPE               ;MASK FOR RMER2
     034260 000405                      BR    8$                ;GO TO 8$ IF NO ERROR
     034262 000240                      NOP                     ;RETURN HERE IF ERROR
     034264 104000                      EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     034266 004736                      JSR   PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     034270 000137 034310               JMP   9$                ;GO TO 9$ IF ERROR
1837 034274                     8$:
```

```
1838 034274 004737 044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     034300 000403                 BR      9$              ;GO TO 9$ IF NO ERROR
     034302 000240                 NOP                     ;RETURN HERE IF ERROR
     034304 104000                 EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     034306 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
1839 034310                    9$:
1840
1841                           ;;*************************************************************************
                               ;*TEST 56        SEEK AT OFFSET
                               ;;*************************************************************************
     034310                    TST56:
     034310 000004                 SCOPE                   ;SCOPE CALL
     034312 000240                 NOP                     ;START OF TEST
     034314 012706 001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     034320 013700 001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     034324 013701 001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     034330 012737 000056 001226   MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1842
1843 034336                    1$:
1844 034336 004737 041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     034342 054130                 .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                           ;CLEAR CONTROLLER & SELECT DEVICE
                                                           ;VERIFY CONTROLLER CLEAR OPERATION
                                                           ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                           ;VERIFY PACK ACKNOWLEDGE
                                                           ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                           ;VERIFY RECALIBRATION
     034344 000404                 BR      2$              ;GO TO 2$ IF NO ERROR
     034346 000240                 NOP                     ;RETURN HERE IF ERROR
     034350 104000                 EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     034352 000137 034634          JMP     10$             ;GO TO 10$ IF ERROR
1845 034356                    2$:
1846 034356 012737 000000 001446   MOV     #0,RMDCO        ;CYLINDER = 0
1847 034364 012737 000000 001420   MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1848 034372 012737 000015 001412   MOV     #OFFSET!GO,RMCS10       ;LOAD OFFSET COMMAND IN BUFFER
1849 034400 012702 001545          MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
1850 034404 112722 000034          MOVB    #RMDC,(R2)+     ;WRITE REGISTER INDEX TABLE
1851 034410 112722 000006          MOVB    #RMDA,(R2)+
1852 034414 112722 000000          MOVB    #RMCS1,(R2)+
1853 034420 112722 000200          MOVB    #200,(R2)+      ;WRITE TERMINATOR
1854 034424 004737 043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     034430 000404                 BR      3$              ;GO TO 3$ IF NO ERROR
     034432 000240                 NOP                     ;RETURN HERE IF ERROR
     034434 104000                 EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     034436 000137 034634          JMP     10$             ;GO TO 10$ IF ERROR
1855 034442 004737 042462      3$: JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
1856 034446 004737 043370          JSR     PC,TIMOUT       ;WAIT FOR OFFSET TO COMPLETE
1857 034452                    4$:
1858 034452 012737 000005 001412   MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
1859 034460 112737 000000 001545   MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
1860 034466 112737 000200 001546   MOVB    #200,PUTINX+1
1861 034474 004737 043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     034500 000404                 BR      5$              ;GO TO 5$ IF NO ERROR
     034502 000240                 NOP                     ;RETURN HERE IF ERROR
     034504 104000                 EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     034506 000137 034634          JMP     10$             ;GO TO 10$ IF ERROR
1862 034512                    5$:
```

```
1863
1864                                    ;WAIT FOR COMMAND TO COMPLETE
     034512  004737  043370                     JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
1865
1866 034516  004737  042546                     JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     034522  000404                             BR      6$              ;GO TO 6$ IF NO ERROR
     034524  000240                             NOP                     ;RETURN HERE IF ERROR
     034526  104000                             EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     034530  000137  034634                     JMP     10$             ;GO TO 10$ IF ERROR
1867 034534                             6$:
1868 034534  004737  043564                     JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     034540  000405                             BR      7$              ;GO TO 7$ IF NO ERROR
     034542  000240                             NOP                     ;RETURN HERE IF ERROR
     034544  104000                             EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     034546  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     034550  000137  034634                     JMP     10$             ;GO TO 10$ IF ERROR
1869 034554                             7$:
1870 034554  004737  051206                     JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
     034560  000405                             BR      8$              ;GO TO 8$ IF NO ERROR
     034562  000240                             NOP                     ;RETURN HERE IF ERROR
     034564  104000                             EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
     034566  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     034570  000137  034634                     JMP     10$             ;GO TO 10$ IF ERROR
1871 034574                             8$:
1872 034574  004737  050562                     JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     034600  115760                             .WORD   NDTMSK          ;MASK FOR RMER1
     034602  000010                             .WORD   DPE             ;MASK FOR RMER2
     034604  000405                             BR      9$              ;GO TO 9$ IF NO ERROR
     034606  000240                             NOP                     ;RETURN HERE IF ERROR
     034610  104000                             EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     034612  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     034614  000137  034634                     JMP     10$             ;GO TO 10$ IF ERROR
1873 034620                             9$:
1874 034620  004737  044416                     JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     034624  000403                             BR      10$             ;GO TO 10$ IF NO ERROR
     034626  000240                             NOP                     ;RETURN HERE IF ERROR
     034630  104000                             EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     034632  004736                             JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
1875 034634                             10$:
1876
1877                                    ;;*******************************************************************
                                        ;*TEST 57        LOOK AHEAD TEST
                                        ;;*******************************************************************
     034634                             TST57:
     034634  000004                             SCOPE                   ;SCOPE CALL
     034636  000240                             NOP                     ;START OF TEST
     034640  012706  001100                     MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     034644  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     034650  013701  001466                     MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     034654  012737  000057  001226             MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1878
1879 034662  004737  052466                     JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     034666  000404                             BR      1$              ;GO TO 1$ IF NO ERROR
     034670  000240                             NOP                     ;RETURN HERE IF ERROR
     034672  104000                             EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     034674  000137  035354                     JMP     20$             ;GO TO 20$ IF ERROR
1880 034700  012703  000037             1$:     MOV     #31.,R3         ;R3 = SAMPLE COUNT FOR 18 BIT MODE
```

```
1881 034704 012737 000000 001444        MOV   #0,RMOFO          ;START WITH 18 BIT MODE
1882 034712 012737 000000 001446        MOV   #0,RMDCO          ;CYLINDER = 0
1883 034720 012737 000000 001420        MOV   #0,RMDAO          ;SEARCH TRACK = 0, SECTOR = 0
1884 034726 012737 000030 001412 2$:    MOV   #SEARCH,RMCS10    ;SEARCH COMMAND
1885 034734 012702 001545               MOV   #PUTINX,R2
1886 034740 112722 000006              MOVB   #RMDA,(R2)+
1887 034744 112722 000034              MOVB   #RMDC,(R2)+
1888 034750 112722 000032              MOVB   #RMOF,(R2)+
1889 034754 112722 000000              MOVB   #RMCS1,(R2)+
1890 034760 112722 000200              MOVB   #200,(R2)+
1891 034764 004737 042462               JSR   PC,GETSTS
1892
1893 034770 004737 043016               JSR   PC,PUT            ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     034774 000404                      BR    3$                ;GO TO 3$ IF NO ERROR
     034776 000240                      NOP                     ;RETURN HERE IF ERROR
     035000 104000                      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     035002 000137 035344               JMP   19$               ;GO TO 19$ IF ERROR
1894 035006                     3$:
1895 035006 012702 000002               MOV   #2,R2             ;DO 2 CLOCK CYCLES
1896 035012 004737 043532        4$:    JSR   PC,STIMER         ;START CLOCK TIMER
1897 035016 032777 000200 144466 5$:    BIT   #BIT7,@CLKADR     ;TIMER DONE??
1898 035024 001774                      BEQ   5$                ;NO!!
1899 035026 005302                      DEC   R2                ;DEC NUMBER OF CYCLES
1900 035030 001370                      BNE   4$                ;CONTINUE IF NOT DONE
1901
1902 035032 012737 000031 001412        MOV   #SEARCH!GO,RMCS10     ;EXECUTE SEARCH COMMAND
1903
1904 035040 004737 043016               JSR   PC,PUT            ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     035044 000404                      BR    6$                ;GO TO 6$ IF NO ERROR
     035046 000240                      NOP                     ;RETURN HERE IF ERROR
     035050 104000                      EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     035052 000137 035344               JMP   19$               ;GO TO 19$ IF ERROR
1905 035056                     6$:
1906 035056 004737 043370               JSR   PC,TIMOUT         ;WAIT FOR COMPLETION
1907
1908 035062 004737 042546               JSR   PC,GET            ;GO READ REGISTER(S) WITH GET SUBROUTINE
     035066 000404                      BR    7$                ;GO TO 7$ IF NO ERROR
     035070 000240                      NOP                     ;RETURN HERE IF ERROR
     035072 104000                      EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     035074 000137 035344               JMP   19$               ;GO TO 19$ IF ERROR
1909 035100                     7$:
1910 035100 004737 043564               JSR   PC,PRIERR         ;GO CHECK FOR PRIMARY ERRORS
     035104 000405                      BR    8$                ;GO TO 8$ IF NO ERROR
     035106 000240                      NOP                     ;RETURN HERE IF ERROR
     035110 104000                      EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     035112 004736                      JSR   PC,@(SP)+         ;GO BACK FOR MORE ERROR CHECKS
     035114 000137 035344               JMP   19$               ;GO TO 19$ IF ERROR
1911 035120                     8$:
1912 035120 012704 104372               MOV   #BUFFER,R4        ;R4 = STORAGE ADDRESS
1913 035124 012705 177777               MOV   #-1,R5            ;R5 = GROSS TIMER
1914 035130 016014 000020        9$:    MOV   RMLA(R0),(R4)     ;STORE RMLA SAMPLE
1915 035134 016002 000020               MOV   RMLA(R0),R2       ;GET ANOTHER LOOK
1916 035140 020214                      CMP   R2,(R4)           ;ARE SAMPLES SAME??
1917 035142 001403                      BEQ   10$               ;YES!!
1918 035144 005305                      DEC   R5                ;TIMEOUT??
1919 035146 001370                      BNE   9$                ;NO - TRY AGAIN
1920 035150 000411                      BR    12$               ;PROGRAM TIMEOUT
```

```
1921 035152 062704 000002          10$:  ADD   #2,R4          ;ADVANCE STORAGE ADDRESS
1922 035156 005303                       DEC   R3             ;ALL SAMPLES TAKEN??
1923 035160 001410                       BEQ   13$            ;YES!!
1924 035162 026002 000020          11$:  CMP   RMLA(R0),R2    ;WAIT FOR CHANGE
1925 035166 001360                       BNE   9$             ;YES!!
1926 035170 005305                       DEC   R5             ;TIMEOUT??
1927 035172 001373                       BNE   11$            ;NO
1928 035174                        12$:
     035174 104352                       EMT   352
1929 035176 000137 035354                JMP   20$
1930
1931                               ;SET 18 BIT SAMPLE PARAMETERS
1932 035202 012702 003500          13$:  MOV   #3500,R2       ;R2 = MAXIMUM SAMPLE
1933 035206 012703 000036                MOV   #30.,R3        ;R3 = NUMBER OF COMPARES
1934 035212 012704 104372                MOV   #BUFFER,R4     ;R4 = BUFFER ADDRESS
1935 035216 032737 010000 001444         BIT   #FMT16,RMOFO   ;IS SAMPLE FOR 18 BIT MODE ?
1936 035224 001404                       BEQ   14$            ;BR IF YES
1937
1938                               ;SET 16 BIT SAMPLE PARAMETERS
1939 035226 012702 003700                MOV   #3700,R2       ;R2 = MAXIMUM SAMPLE
1940 035232 012703 000040                MOV   #32.,R3        ;R3 = NUMBER OF COMPARES
1941 035236 012405                 14$:  MOV   (R4)+,R5       ;GET A SAMPLE AND INCREMENT
1942 035240 062705 000100                ADD   #100,R5        ;FOR EXPECTED VALUE OF NEXT
1943 035244 020205                       CMP   R2,R5          ;SHOULD NEXT BE 0 ?
1944 035246 103001                       BHIS  15$            ;NO!!
1945 035250 005005                       CLR   R5             ;YES - CHANGE EXPECTED
1946 035252 020514                 15$:  CMP   R5,(R4)        ;IS NEXT SAMPLE CORRECT??
1947 035254 001406                       BEQ   16$            ;YES!!
1948 035256 010537 001140                MOV   R5,$GDDAT      ;EXPECTED VALUE
1949 035262 011437 001142                MOV   (R4),$BDDAT    ;RECEIVED VALUE
1950 035266 104353                       EMT   353
1951 035270 000425                       BR    19$
1952 035272 005303                 16$:  DEC   R3             ;ALL SAMPLES CHECKED??
1953 035274 001360                       BNE   14$            ;NO - TEST NEXT SAMPLE
1954
1955 035276 032737 010000 001444         BIT   #FMT16,RMOFO   ;TEST DONE FOR 16 BIT MODE ?
1956 035304 001004                       BNE   17$            ;BR IF YES
1957 035306 012737 010000 001444         MOV   #FMT16,RMOFO   ;SET 16 BIT MODE AND
1958 035314 000407                       BR    18$            ;DO AGAIN FOR 16 BIT MODE
1959
1960 035316 032737 001000 001444   17$:  BIT   #SSEI,RMOFO    ;TEST FOR 16 BIT MODE W/ SSEI SET ?
1961 035324 001007                       BNE   19$            ;BR IF YES
1962 035326 052737 001000 001444         BIS   #SSEI,RMOFO    ;SETUP FOR 16 BIT MODE W/ SSEI AND
1963 035334 012703 000041          18$:  MOV   #33.,R3        ;SET SAMPLE COUNT.
1964 035340 000137 034726                JMP   2$             ;DO AGAIN FOR 16 BIT MODE W/ SSEI
1965
1966 035344                        19$:
     035344 012637 000006                MOV   (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1967 035350 012637 000004                MOV   (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1968 035354                        20$:
1969
1970
                                   ;;*************************************************************************
                                   ;*TEST 60       SEARCH ON CYLINDER
                                   ;;*************************************************************************
     035354                        TST60:
     035354 000004                       SCOPE                ;SCOPE CALL
     035356 000240                       NOP                  ;START OF TEST
```

```
                035360  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                035364  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                035370  013701  001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                035374  012737  000060  001226  MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1971
1972            035402  004737  041532          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                035406  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                        ;VERIFY PACK ACKNOWLEDGE
                                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                        ;VERIFY RECALIBRATION
                035410  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
                035412  000240                  NOP                     ;RETURN HERE IF ERROR
                035414  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                035416  000137  035776          JMP     12$             ;GO TO 12$ IF ERROR
1973            035422                  1$:
1974            035422  012703  000037          MOV     #31.,R3         ;R3 = MAXIMUM SECTOR ADDRESS
1975            035426  012737  000000  001446  MOV     #0,RMDCO        ;CYLINDER = 0
1976            035434  012737  000000  001420  MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
1977            035442  012737  011000  001444  MOV     #FMT16!SSEI,RMOFO       ;SET 16 BIT MODE W/ SSEI
1978            035450  012737  000005  001412  MOV     #SEEK!GO,RMCS10 ;SEEK COMMAND
1979            035456  012702  001545          MOV     #PUTINX,R2      ;SETUP REGISTER INDEX TABLE
1980            035462  112722  000006          MOVB    #RMDA,(R2)+     ;FOR SEEK TO CYLINDER 0
1981            035466  112722  000034          MOVB    #RMDC,(R2)+
1982            035472  112722  000032          MOVB    #RMOF,(R2)+
1983            035476  112722  000000          MOVB    #RMCS1,(R2)+
1984            035502  112722  000200          MOVB    #200,(R2)+
1985            035506  004737  042462          JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
1986            035512  004737  043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                035516  000404                  BR      2$              ;GO TO 2$ IF NO ERROR
                035520  000240                  NOP                     ;RETURN HERE IF ERROR
                035522  104000                  EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                035524  000137  035776          JMP     12$             ;GO TO 12$ IF ERROR
1987            035530  004737  043370  2$:     JSR     PC,TIMOUT
1988
1989            035534  004737  042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                035540  000404                  BR      3$              ;GO TO 3$ IF NO ERROR
                035542  000240                  NOP                     ;RETURN HERE IF ERROR
                035544  104000                  EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                035546  000137  035776          JMP     12$             ;GO TO 12$ IF ERROR
1990            035552                  3$:
1991            035552  004737  043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                035556  000405                  BR      4$              ;GO TO 4$ IF NO ERROR
                035560  000240                  NOP                     ;RETURN HERE IF ERROR
                035562  104000                  EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                035564  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                035566  000137  035776          JMP     12$             ;GO TO 12$ IF ERROR
1992            035572                  4$:
1993            035572  004737  051206          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
                035576  000405                  BR      5$              ;GO TO 5$ IF NO ERROR
                035600  000240                  NOP                     ;RETURN HERE IF ERROR
                035602  104000                  EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                035604  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                035606  000137  035776          JMP     12$             ;GO TO 12$ IF ERROR
1994            035612  012737  000031  001412  5$:     MOV     #SEARCH!GO,RMCS10       ;STORE SEARCH COMMAND
```

```
1995 035620 004737 043016                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     035624 000404                       BR      6$              ;GO TO 6$ IF NO ERROR
     035626 000240                       NOP                     ;RETURN HERE IF ERROR
     035630 104000                       EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     035632 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
1996 035636                         6$:
1997 035636 004737 043370                JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
1998
1999 035642 004737 042546                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     035646 000404                       BR      7$              ;GO TO 7$ IF NO ERROR
     035650 000240                       NOP                     ;RETURN HERE IF ERROR
     035652 104000                       EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     035654 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
2000 035660                         7$:
2001 035660 004737 043564                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     035664 000405                       BR      8$              ;GO TO 8$ IF NO ERROR
     035666 000240                       NOP                     ;RETURN HERE IF ERROR
     035670 104000                       EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     035672 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     035674 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
2002 035700                         8$:
2003 035700 004737 056624                JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
     035704 000405                       BR      9$              ;GO TO 9$ IF NO ERROR
     035706 000240                       NOP                     ;RETURN HERE IF ERROR
     035710 104000                       EMT                     ;ERROR # DEFINED BY SCHSTS SUBROUTINE
     035712 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     035714 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
2004 035720                         9$:
2005 035720 004737 050562                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     035724 115760                       .WORD   NDTMSK          ;MASK FOR RMER1
     035726 000010                       .WORD   DPE             ;MASK FOR RMER2
     035730 000405                       BR      10$             ;GO TO 10$ IF NO ERROR
     035732 000240                       NOP                     ;RETURN HERE IF ERROR
     035734 104000                       EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     035736 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     035740 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
2006 035744                         10$:
2007 035744 004737 044416                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     035750 000405                       BR      11$             ;GO TO 11$ IF NO ERROR
     035752 000240                       NOP                     ;RETURN HERE IF ERROR
     035754 104000                       EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     035756 004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     035760 000137 035776                JMP     12$             ;GO TO 12$ IF ERROR
2008 035764                         11$:
2009 035764 005237 001420                INC     RMDAO           ;INCREMENT SECTOR ADDRESS
2010 035770 020337 001420                CMP     R3,RMDAO        ;DONE ALL SECTORS??
2011 035774 103306                       BHIS    5$              ;NO!!
2012 035776                         12$:
2013
2014                                ;;*****************************************************************
                                    ;*TEST 61        SEARCH OFF CYLINDER
                                    ;;*****************************************************************
     035776                         TST61:
     035776 000004                       SCOPE                   ;SCOPE CALL
     036000 000240                       NOP                     ;START OF TEST
     036002 012706 001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     036006 013700 001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
```

```
                036012  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                036016  012737  000061  001226          MOV     #61.,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
2015
2016            036024  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                036030  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                036032  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                036034  000240                          NOP                     ;RETURN HERE IF ERROR
                036036  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                036040  000137  036440                  JMP     13$             ;GO TO 13$ IF ERROR
2017            036044                          1$:
2018            036044  012737  000000  001420          MOV     #0,RMDAO        ;START AT TRACK = 0, SECTOR = 0
2019            036052  012737  011000  001444          MOV     #FMT16!SSEI,RMOFO       ;SET 16 BIT MODE W/ SSEI
2020            036060  012703  000427                  MOV     #279.,R3        ;R3 = SEARCH CYLINDER
2021            036064  012705  000430                  MOV     #280.,R5        ;R5 = SEEK CYLINDER
2022            036070  012737  000005  001412  2$:     MOV     #SEEK!GO,RMCS10 ;LOAD SEEK COMMAND
2023            036076  010537  001446                  MOV     R5,RMDCO        ;LOAD CYLINDER ADDRESS
2024            036102  012702  001545                  MOV     #PUTINX,R2      ;R2 POINTS TO REGISTER TABLE
2025            036106  112722  000032                  MOVB    #RMOF,(R2)+     ;SETUP REGISTER INDEX TABLE
2026            036112  112722  000034                  MOVB    #RMDC,(R2)+     ;FOR SEEK COMMAND
2027            036116  112722  000006                  MOVB    #RMDA,(R2)+
2028            036122  112722  000000                  MOVB    #RMCS1,(R2)+
2029            036126  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
2030            036132  004737  042462                  JSR     PC,GETSTS       ;SETUP FOR STATUS
2031            036136  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                036142  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                036144  000240                          NOP                     ;RETURN HERE IF ERROR
                036146  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                036150  000137  036440                  JMP     13$             ;GO TO 13$ IF ERROR
2032            036154                          3$:
2033            036154  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR SEEK TO COMPLETE
2034
2035            036160  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                036164  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                036166  000240                          NOP                     ;RETURN HERE IF ERROR
                036170  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                036172  000137  036440                  JMP     13$             ;GO TO 13$ IF ERROR
2036            036176                          4$:
2037            036176  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                036202  000405                          BR      5$              ;GO TO 5$ IF NO ERROR
                036204  000240                          NOP                     ;RETURN HERE IF ERROR
                036206  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                036210  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036212  000137  036440                  JMP     13$             ;GO TO 13$ IF ERROR
2038            036216                          5$:
2039            036216  004737  051206                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
                036222  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
                036224  000240                          NOP                     ;RETURN HERE IF ERROR
                036226  104000                          EMT                     ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                036230  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036232  000137  036440                  JMP     13$             ;GO TO 13$ IF ERROR
2040            036236                          6$:
```

```
T61     SEARCH OFF CYLINDER

2041 036236 010337 001446          MOV     R3,RMDCO        ;LOAD CYLINDER ADDRESS
2042 036242 012737 000031 001412   MOV     #SEARCH!GO,RMCS1O       ;LOAD SEARCH COMMAND
2043 036250 004737 043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     036254 000404                 BR      7$              ;GO TO 7$ IF NO ERROR
     036256 000240                 NOP                     ;RETURN HERE IF ERROR
     036260 104000                 EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     036262 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2044 036266                    7$:
2045 036266 004737 043370          JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE
2046
2047 036272 004737 042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     036276 000404                 BR      8$              ;GO TO 8$ IF NO ERROR
     036300 000240                 NOP                     ;RETURN HERE IF ERROR
     036302 104000                 EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     036304 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2048 036310                    8$:
2049 036310 004737 043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     036314 000405                 BR      9$              ;GO TO 9$ IF NO ERROR
     036316 000240                 NOP                     ;RETURN HERE IF ERROR
     036320 104000                 EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     036322 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     036324 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2050 036330                    9$:
2051 036330 004737 056624          JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
     036334 000405                 BR      10$             ;GO TO 10$ IF NO ERROR
     036336 000240                 NOP                     ;RETURN HERE IF ERROR
     036340 104000                 EMT                     ;ERROR # DEFINED BY SCHSTS SUBROUTINE
     036342 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     036344 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2052 036350                   10$:
2053 036350 004737 050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     036354 115760                 .WORD   NDTMSK          ;MASK FOR RMER1
     036356 000010                 .WORD   DPE             ;MASK FOR RMER2
     036360 000405                 BR      11$             ;GO TO 11$ IF NO ERROR
     036362 000240                 NOP                     ;RETURN HERE IF ERROR
     036364 104000                 EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     036366 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     036370 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2054 036374                   11$:
2055 036374 004737 044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     036400 000405                 BR      12$             ;GO TO 12$ IF NO ERROR
     036402 000240                 NOP                     ;RETURN HERE IF ERROR
     036404 104000                 EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     036406 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     036410 000137 036440          JMP     13$             ;GO TO 13$ IF ERROR
2056 036414                   12$:
2057 036414 005303                 DEC     R3              ;DECREMENT SEARCH CYLINDER
2058 036416 005205                 INC     R5              ;INCREMENT SEEK CYLINDER
2059 036420 005237 001420          INC     RMDA0           ;INCREMENT SECTOR ADDRESS
2060 036424 023727 001420 000037   CMP     RMDA0,#31.      ;DONE ALL SECTORS??
2061 036432 101002                 BHI     13$             ;BR IF YES
2062 036434 000137 036070          JMP     2$              ;DO NEXT SECTOR
2063 036440                   13$:
2064
2065
                                ;;*****************************************************************
                                ;*TEST 62        SEARCH INVALID SECTOR
                                ;;*****************************************************************
```

```
                036440                          TST62:
                036440  000004                          SCOPE                   ;SCOPE CALL
                036442  000240                          NOP                     ;START OF TEST
                036444  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                036450  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                036454  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                036460  012737  000062  001226          MOV     #62,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   2066
   2067         036466  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                036472  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                036474  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                036476  000240                          NOP                     ;RETURN HERE IF ERROR
                036500  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                036502  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
   2068         036506                          1$:
   2069         036506  012737  000000  001444          MOV     #0,RMOFO        ;SET 18 BIT FORMAT
   2070         036514  012737  000000  001446          MOV     #0,RMDCO        ;CYLINDER = 0
   2071         036522  012737  000036  001420          MOV     #30.,RMDAO      ;FIRST INVALID SECTOR = 30.
   2072         036530  012737  000031  001412          MOV     #SEARCH!GO,RMCS10       ;SEARCH COMMAND
   2073         036536  012702  001545                  MOV     #PUTINX,R2      ;WRITE REGISTER TABLE FOR
   2074         036542  112722  000034                  MOVB    #RMDC,(R2)+     ;COMMAND
   2075         036546  112722  000032                  MOVB    #RMOF,(R2)+
   2076         036552  112722  000006                  MOVB    #RMDA,(R2)+
   2077         036556  112722  000000                  MOVB    #RMCS1,(R2)+
   2078         036562  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
   2079         036566  004737  042462                  JSR     PC,GETSTS       ;SETUP STATUS FETCH
   2080         036572                          2$:
   2081         036572  004737  052466                  JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
                036576  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                036600  000240                          NOP                     ;RETURN HERE IF ERROR
                036602  104000                          EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
                036604  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
   2082         036610                          3$:
   2083         036610  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                036614  000404                          BR      4$              ;GO TO 4$ IF NO ERROR
                036616  000240                          NOP                     ;RETURN HERE IF ERROR
                036620  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                036622  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
   2084         036626                          4$:
   2085         036626  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
   2086
   2087         036632  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                036636  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                036640  000240                          NOP                     ;RETURN HERE IF ERROR
                036642  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                036644  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
   2088         036650                          5$:
   2089         036650  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                036654  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
                036656  000240                          NOP                     ;RETURN HERE IF ERROR
                036660  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
```

```
                036662  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036664  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
2090            036670                          6$:
2091            036670  004737  056624                  JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
                036674  000405                          BR      7$              ;GO TO 7$ IF NO ERROR
                036676  000240                          NOP                     ;RETURN HERE IF ERROR
                036700  104000                          EMT                     ;ERROR # DEFINED BY SCHSTS SUBROUTINE
                036702  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036704  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
2092            036710                          7$:
2093            036710  004737  060210                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
                036714  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
                036716  000240                          NOP                     ;RETURN HERE IF ERROR
                036720  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
                036722  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036724  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
2094            036730                          8$:
2095            036730  004737  050562                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                036734  117760                          .WORD   IAE!NDTMSK              ;MASK FOR RMER1
                036736  000010                          .WORD   DPE             ;MASK FOR RMER2
                036740  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
                036742  000240                          NOP                     ;RETURN HERE IF ERROR
                036744  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                036746  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036750  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
2096            036754                          9$:
2097            036754  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                036760  000405                          BR      10$             ;GO TO 10$ IF NO ERROR
                036762  000240                          NOP                     ;RETURN HERE IF ERROR
                036764  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                036766  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                036770  000137  037040                  JMP     12$             ;GO TO 12$ IF ERROR
2098            036774                          10$:
2099            036774  005237  001420                  INC     RMDA0           ;INCREMENT SECTOR ADDRESS
2100            037000  023727  001420  000100          CMP     RMDA0,#64.      ;DONE YET ?
2101            037006  101412                          BLOS    11$             ;NO - TEST NEXT SECTOR
2102
2103            037010  032737  010000  001444          BIT     #FMT16,RMOF0    ;TEST FOR 16 BIT MODE YET ?
2104            037016  001010                          BNE     12$             ;BR IF YES
2105            037020  012737  010000  001444          MOV     #FMT16,RMOF0    ;SET 16 BIT MODE
2106            037026  012737  000037  001420          MOV     #31.,RMDA0      ;SET INVALID SECTOR = 31.
2107            037034  000137  036572                  JMP     2$              ;TEST NEXT SECTOR
2108            037040                          11$:
2109                                            12$:
2110
                                                ;;*****************************************************************
                                                ;*TEST 63        SEARCH INVALID TRACK
                                                ;;*****************************************************************
                037040                          TST63:
                037040  000004                          SCOPE                   ;SCOPE CALL
                037042  000240                          NOP                     ;START OF TEST
                037044  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                037050  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                037054  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                037060  012737  000063  001226          MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2111
2112            037066  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                037072  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
```

```
                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                        ;VERIFY PACK ACKNOWLEDGE
                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                        ;VERIFY RECALIBRATION
 037074  000404                       BR      1$       ;GO TO 1$ IF NO ERROR
 037076  000240                       NOP              ;RETURN HERE IF ERROR
 037100  104000                       EMT              ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 037102  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
2113 037106                    1$:
2114 037106  012737  000031  001412   MOV     #SEARCH!GO,RMCS1O       ;SEARCH COMMAND
2115 037114  012737  000000  001446   MOV     #0,RMDCO        ;CYLINDER = 0
2116 037122  013737  001334  001420   MOV     LSTRK,RMDAO     ;LAST TRACK, SECTOR = 0
2117 037130  105237  001421           INCB    RMDAO+1         ;SETUP FIRST INVALID TRACK
2118 037134  012737  010000  001444   MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
2119 037142  012702  001545           MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
2120 037146  112722  000006           MOVB    #RMDA,(R2)+
2121 037152  112722  000034           MOVB    #RMDC,(R2)+
2122 037156  112722  000032           MOVB    #RMOF,(R2)+
2123 037162  112722  000000           MOVB    #RMCS1,(R2)+
2124 037166  112722  000200           MOVB    #200,(R2)+      ;TERMINATE TABLE
2125 037172  004737  042462           JSR     PC,GETSTS
2126 037176                    2$:
2127 037176  004737  052466           JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
 037202  000404                       BR      3$       ;GO TO 3$ IF NO ERROR
 037204  000240                       NOP              ;RETURN HERE IF ERROR
 037206  104000                       EMT              ;ERROR NUMBER DEFINED BY SUBROUTINE
 037210  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
2128 037214                    3$:
2129 037214  004737  043016           JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
 037220  000404                       BR      4$       ;GO TO 4$ IF NO ERROR
 037222  000240                       NOP              ;RETURN HERE IF ERROR
 037224  104000                       EMT              ;ERROR # DEFINED BY PUT SUBROUTINE
 037226  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
2130 037232                    4$:
2131 037232  004737  043370           JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
2132
2133 037236  004737  042546           JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
 037242  000404                       BR      5$       ;GO TO 5$ IF NO ERROR
 037244  000240                       NOP              ;RETURN HERE IF ERROR
 037246  104000                       EMT              ;ERROR # DEFINED BY GET SUBROUTINE
 037250  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
2134 037254                    5$:
2135 037254  004737  043564           JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 037260  000405                       BR      6$       ;GO TO 6$ IF NO ERROR
 037262  000240                       NOP              ;RETURN HERE IF ERROR
 037264  104000                       EMT              ;ERROR # DEFINED BY PRIERR SUBROUTINE
 037266  004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 037270  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
2136 037274                    6$:
2137 037274  004737  056624           JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
 037300  000405                       BR      7$       ;GO TO 7$ IF NO ERROR
 037302  000240                       NOP              ;RETURN HERE IF ERROR
 037304  104000                       EMT              ;ERROR # DEFINED BY SCHSTS SUBROUTINE
 037306  004736                       JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 037310  000137  037420               JMP     11$      ;GO TO 11$ IF ERROR
```

```
2138 037314                          7$:
2139 037314  004737  060210                  JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
     037320  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
     037322  000240                          NOP                     ;RETURN HERE IF ERROR
     037324  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
     037326  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037330  000137  037420                  JMP     11$             ;GO TO 11$ IF ERROR
2140 037334                          8$:
2141 037334  004737  050562                  JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     037340  117760                          .WORD   NDTMSK!IAE              ;MASK FOR RMER1
     037342  000010                          .WORD   DPE             ;MASK FOR RMER2
     037344  000405                          BR      9$              ;GO TO 9$ IF NO ERROR
     037346  000240                          NOP                     ;RETURN HERE IF ERROR
     037350  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     037352  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037354  000137  037420                  JMP     11$             ;GO TO 11$ IF ERROR
2142 037360                          9$:
2143 037360  004737  044416                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     037364  000405                          BR      10$             ;GO TO 10$ IF NO ERROR
     037366  000240                          NOP                     ;RETURN HERE IF ERROR
     037370  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     037372  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037374  000137  037420                  JMP     11$             ;GO TO 11$ IF ERROR
2144 037400                          10$:
2145 037400  105237  001421                  INCB    RMDA0+1         ;INCREMENT TRACK ADDRESS
2146 037404  123727  001421  000200          CMPB    RMDA0+1,#128.   ;DONE ALL TRACKS ?
2147 037412  101002                          BHI     11$             ;BR IF YES
2148 037414  000137  037176                  JMP     2$              ;TEST NEXT TRACK
2149 037420                          11$:
2150
2151                                 ;;*****************************************************************
                                     ;*TEST 64       SEARCH INVALID CYLINDER
                                     ;;*****************************************************************
     037420                          TST64:
     037420  000004                          SCOPE                   ;SCOPE CALL
     037422  000240                          NOP                     ;START OF TEST
     037424  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     037430  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     037434  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
     037440  012737  000064  001226          MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2152
2153 037446  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
     037452  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                     ;CLEAR CONTROLLER & SELECT DEVICE
                                                                     ;VERIFY CONTROLLER CLEAR OPERATION
                                                                     ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                     ;VERIFY PACK ACKNOWLEDGE
                                                                     ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                     ;VERIFY RECALIBRATION
     037454  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
     037456  000240                          NOP                     ;RETURN HERE IF ERROR
     037460  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
     037462  000137  037774                  JMP     11$             ;GO TO 11$ IF ERROR
2154 037466                          1$:
2155 037466  012737  000031  001412          MOV     #SEARCH!GO,RMCS1O       ;SEARCH COMMAND
2156 037474  012737  001061  001446          MOV     #561.,RMDCO     ;START AT FIRST INVALID CYLINDER
2157 037502  012737  000000  001420          MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
```

```
2158 037510 012737 010000 001444          MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
2159 037516 012702 001545                 MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
2160 037522 112722 000032                 MOVB    #RMOF,(R2)+
2161 037526 112722 000034                 MOVB    #RMDC,(R2)+
2162 037532 112722 000006                 MOVB    #RMDA,(R2)+
2163 037536 112722 000000                 MOVB    #RMCS1,(R2)+
2164 037542 112722 000200                 MOVB    #200,(R2)+
2165 037546 004737 042462                 JSR     PC,GETSTS       ;SETUP STATUS FETCH
2166 037552                         2$:
2167 037552 004737 052466                 JSR     PC,CNTCLR       ;GO ISSUE CONTROLLER CLEAR
     037556 000404                        BR      3$              ;GO TO 3$ IF NO ERROR
     037560 000240                        NOP                     ;RETURN HERE IF ERROR
     037562 104000                        EMT                     ;ERROR NUMBER DEFINED BY SUBROUTINE
     037564 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2168 037570                         3$:
2169 037570 004737 043016                 JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     037574 000404                        BR      4$              ;GO TO 4$ IF NO ERROR
     037576 000240                        NOP                     ;RETURN HERE IF ERROR
     037600 104000                        EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     037602 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2170 037606                         4$:
2171 037606 004737 043370                 JSR     PC,TIMOUT
2172
2173 037612 004737 042546                 JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     037616 000404                        BR      5$              ;GO TO 5$ IF NO ERROR
     037620 000240                        NOP                     ;RETURN HERE IF ERROR
     037622 104000                        EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     037624 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2174 037630                         5$:
2175 037630 004737 043564                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     037634 000405                        BR      6$              ;GO TO 6$ IF NO ERROR
     037636 000240                        NOP                     ;RETURN HERE IF ERROR
     037640 104000                        EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     037642 004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037644 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2176 037650                         6$:
2177 037650 004737 056624                 JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
     037654 000405                        BR      7$              ;GO TO 7$ IF NO ERROR
     037656 000240                        NOP                     ;RETURN HERE IF ERROR
     037660 104000                        EMT                     ;ERROR # DEFINED BY SCHSTS SUBROUTINE
     037662 004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037664 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2178 037670                         7$:
2179 037670 004737 060210                 JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
     037674 000405                        BR      8$              ;GO TO 8$ IF NO ERROR
     037676 000240                        NOP                     ;RETURN HERE IF ERROR
     037700 104000                        EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
     037702 004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     037704 000137 037774                 JMP     11$             ;GO TO 11$ IF ERROR
2180 037710                         8$:
2181 037710 004737 050562                 JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     037714 117760                        .WORD   IAE!NDTMSK              ;MASK FOR RMER1
     037716 000010                        .WORD   DPE             ;MASK FOR RMER2
     037720 000405                        BR      9$              ;GO TO 9$ IF NO ERROR
     037722 000240                        NOP                     ;RETURN HERE IF ERROR
     037724 104000                        EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     037726 004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
037730  000137  037774                  JMP     11$             ;GO TO 11$ IF ERROR
2182 037734                     9$:
2183 037734  004737  044416             JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
037740  000405                  BR      10$             ;GO TO 10$ IF NO ERROR
037742  000240                  NOP                     ;RETURN HERE IF ERROR
037744  104000                  EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
037746  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
037750  000137  037774             JMP     11$             ;GO TO 11$ IF ERROR
2184 037754                     10$:
2185 037754  005237  001446             INC     RMDCO           ;INCREMENT CYLINDER ADDRESS
2186 037760  023727  001446  002000       CMP     RMDCO,#1024.    ;DONE ALL CYLINDERS ?
2187 037766  103002                  BHIS    11$             ;BR IF YES
2188 037770  000137  037552             JMP     2$              ;TEST NEXT INVALID CYLINDER
2189 037774                     11$:
2190
2191                            ;;*******************************************************************
                                ;*TEST 65        IVC SEARCH TEST
                                ;;*******************************************************************
037774                          TST65:
037774  000004                  SCOPE                   ;SCOPE CALL
037776  000240                  NOP                     ;START OF TEST
040000  012706  001100             MOV     #STACK,SP       ;INITIALIZE STACK POINTER
040004  013700  001276             MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
040010  013701  001466             MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
040014  012737  000065  001226       MOV     #65,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2192
2193 040022  004737  041532             JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
040026  054130                  .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                        ;CLEAR CONTROLLER & SELECT DEVICE
                                                        ;VERIFY CONTROLLER CLEAR OPERATION
                                                        ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                        ;VERIFY PACK ACKNOWLEDGE
                                                        ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                        ;VERIFY RECALIBRATION
040030  000404                  BR      1$              ;GO TO 1$ IF NO ERROR
040032  000240                  NOP                     ;RETURN HERE IF ERROR
040034  104000                  EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
040036  000137  040354             JMP     8$              ;GO TO 8$ IF ERROR
2194 040042                     1$:
2195 040042  112737  000024  001545       MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
040050  112737  000200  001546       MOVB    #200,PUTINX+1   ;SET TERMINATOR BYTE
040056  012737  000001  001436       MOV     #DMD,RMMR10     ;SET RMMR1 OUTPUT BUFFER = DMD
040064  004737  043016             JSR     PC,PUT          ;GO WRITE RMMR1 VIA PUT SUBROUTINE
040070  000404                  BR      2$              ;GO TO 2$ IF NO ERROR
040072  000240                  NOP                     ;RETURN HERE IF ERROR
040074  104000                  EMT                     ;ERROR DEFINED BY PUT SUBROUTINE
040076  000137  040354             JMP     8$              ;GO TO 8$ IF ERROR
2196 040102                     2$:
2197 040102  012737  000000  001436       MOV     #0,RMMR10       ;RESET DIAGNOSTIC MODE
2198 040110  012737  000000  001446       MOV     #0,RMDCO        ;CYLINDER 0
2199 040116  012737  000000  001420       MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
2200 040124  012737  000000  001444       MOV     #0,RMOFO        ;18 BIT FORMAT
2201 040132  012737  000031  001412       MOV     #SEARCH!GO,RMCS10       ;SEARCH COMMAND
2202 040140  012702  001545             MOV     #PUTINX,R2      ;LOAD INDEX TABLE
2203 040144  112722  000024             MOVB    #RMMR1,(R2)+
2204 040150  112722  000034             MOVB    #RMDC,(R2)+
2205 040154  112722  000006             MOVB    #RMDA,(R2)+
```

```
2206 040160 112722 000032          MOVB    #RMOF,(R2)+
2207 040164 112722 000000          MOVB    #RMCS1,(R2)+
2208 040170 112722 000200          MOVB    #200,(R2)+
2209 040174 004737 042462          JSR     PC,GETSTS
2210 040200 004737 043016          JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
     040204 000404                 BR      3$              ;GO TO 3$ IF NO ERROR
     040206 000240                 NOP                     ;RETURN HERE IF ERROR
     040210 104000                 EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
     040212 000137 040354          JMP     8$              ;GO TO 8$ IF ERROR
2211 040216                    3$:
2212 040216 004737 043370          JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
2213
2214 040222 004737 042546          JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
     040226 000404                 BR      4$              ;GO TO 4$ IF NO ERROR
     040230 000240                 NOP                     ;RETURN HERE IF ERROR
     040232 104000                 EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
     040234 000137 040354          JMP     8$              ;GO TO 8$ IF ERROR
2215 040240                    4$:
2216 040240 004737 043564          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
     040244 000405                 BR      5$              ;GO TO 5$ IF NO ERROR
     040246 000240                 NOP                     ;RETURN HERE IF ERROR
     040250 104000                 EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
     040252 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     040254 000137 040354          JMP     8$              ;GO TO 8$ IF ERROR
2217 040260                    5$:
2218 040260 032737 010000 001400   BIT     #IVC,RMER2I     ;DID IVC SET??
2219 040266 001012                 BNE     6$              ;YES!!
2220 040270 012737 010000 001140   MOV     #IVC,$GDDAT     ;EXPECTED STATUS
2221 040276 013737 001400 001142   MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
2222 040304 042737 167777 001142   BIC     #^CIVC,$BDDAT
2223 040312 104260                 EMT     260
2224 040314                    6$:
2225 040314 004737 050562          JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
     040320 115760                 .WORD   NDTMSK          ;MASK FOR RMER1
     040322 010010                 .WORD   IVC!DPE         ;MASK FOR RMER2
     040324 000405                 BR      7$              ;GO TO 7$ IF NO ERROR
     040326 000240                 NOP                     ;RETURN HERE IF ERROR
     040330 104000                 EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
     040332 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
     040334 000137 040354          JMP     8$              ;GO TO 8$ IF ERROR
2226 040340                    7$:
2227 040340 004737 044416          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
     040344 000403                 BR      8$              ;GO TO 8$ IF NO ERROR
     040346 000240                 NOP                     ;RETURN HERE IF ERROR
     040350 104000                 EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
     040352 004736                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
2228 040354                    8$:
2229
2230                           ;;*****************************************************************
                               ;*TEST 66        ABORT SEARCH TEST
                               ;;*****************************************************************
     040354                    TST66:
     040354 000004                 SCOPE                   ;SCOPE CALL
     040356 000240                 NOP                     ;START OF TEST
     040360 012706 001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
     040364 013700 001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     040370 013701 001466          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
                040374  012737  000066  001226          MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2231
2232            040402  004737  041532                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                040406  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                                ;VERIFY RECALIBRATION
                040410  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
                040412  000240                          NOP                     ;RETURN HERE IF ERROR
                040414  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                040416  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2233            040422                          1$:
2234            040422  012737  040000  001426          MOV     #UNS,RMER10     ;SET UNSAFE ERROR
2235            040430  012737  000031  001412          MOV     #SEARCH!GO,RMCS10       ;SEARCH COMMAND
2236            040436  012737  000000  001446          MOV     #0,RMDC0        ;CYLINDER 0
2237            040444  012737  000000  001420          MOV     #0,RMDA0        ;SECTOR = 0, TRACK = 0
2238            040452  012702  001545                  MOV     #PUTINX,R2      ;WRITE REGISTER INDEX TABLE
2239            040456  112722  000034                  MOVB    #RMDC,(R2)+
2240            040462  112722  000006                  MOVB    #RMDA,(R2)+
2241            040466  112722  000014                  MOVB    #RMER1,(R2)+
2242            040472  112722  000000                  MOVB    #RMCS1,(R2)+
2243            040476  112722  000200                  MOVB    #200,(R2)+      ;TERMINATE TABLE
2244            040502  004737  042462                  JSR     PC,GETSTS       ;SETUP FOR STATUS
2245            040506  004737  043016                  JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                040512  000404                          BR      2$              ;GO TO 2$ IF NO ERROR
                040514  000240                          NOP                     ;RETURN HERE IF ERROR
                040516  104000                          EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                040520  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2246            040524                          2$:
2247            040524  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                040530  000404                          BR      3$              ;GO TO 3$ IF NO ERROR
                040532  000240                          NOP                     ;RETURN HERE IF ERROR
                040534  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                040536  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2248            040542                          3$:
2249            040542  032737  020000  001350          BIT     #PIP,RMDSI      ;IS PIP SET??
2250            040550  001412                          BEQ     4$              ;NO!!
2251            040552  012737  000000  001140          MOV     #0,$GDDAT       ;EXPECTED STATUS
2252            040560  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
2253            040566  042737  157777  001142          BIC     #^CPIP,$BDDAT
2254            040574  104261                          EMT     261
2255            040576                          4$:
2256            040576  004737  043370                  JSR     PC,TIMOUT       ;WAIT FOR GO TO RESET
2257
2258            040602  004737  042546                  JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                040606  000404                          BR      5$              ;GO TO 5$ IF NO ERROR
                040610  000240                          NOP                     ;RETURN HERE IF ERROR
                040612  104000                          EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                040614  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2259            040620                          5$:
2260            040620  004737  043564                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                040624  000405                          BR      6$              ;GO TO 6$ IF NO ERROR
                040626  000240                          NOP                     ;RETURN HERE IF ERROR
                040630  104000                          EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
```

```
040632  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
040634  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2261  040640                    6$:
2262  040640  004737  060210            JSR     PC,STCDRVSTS    ;GO CHECK FOR CHANGES IN DRIVE STATUS
040644  000405                          BR      7$              ;GO TO 7$ IF NO ERROR
040646  000240                          NOP                     ;RETURN HERE IF ERROR
040650  104000                          EMT                     ;ERROR # DEFINED BY STCDRVSTS SUBROUTINE
040652  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
040654  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2263  040660                    7$:
2264  040660  004737  050562            JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
040664  155760                          .WORD   NDTMSK!UNS              ;MASK FOR RMER1
040666  000010                          .WORD   DPE             ;MASK FOR RMER2
040670  000405                          BR      8$              ;GO TO 8$ IF NO ERROR
040672  000240                          NOP                     ;RETURN HERE IF ERROR
040674  104000                          EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
040676  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
040700  000137  040720                  JMP     9$              ;GO TO 9$ IF ERROR
2265  040704                    8$:
2266  040704  004737  044416            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
040710  000403                          BR      9$              ;GO TO 9$ IF NO ERROR
040712  000240                          NOP                     ;RETURN HERE IF ERROR
040714  104000                          EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
040716  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
2267  040720                    9$:
2268
2269                              ;;*****************************************************************
                                  ;*TEST 67        SEARCH AT OFFSET
                                  ;;*****************************************************************
040720                            TST67:
040720  000004                          SCOPE                   ;SCOPE CALL
040722  000240                          NOP                     ;START OF TEST
040724  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
040730  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
040734  013701  001466                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
040740  012737  000067  001226          MOV     #67,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2270
2271  040746  004737  041532            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
040752  054130                          .WORD   054130          ;TASK DESCRIPTOR AS FOLLOWS:
                                                                ;CLEAR CONTROLLER & SELECT DEVICE
                                                                ;VERIFY CONTROLLER CLEAR OPERATION
                                                                ;PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                                                ;VERIFY PACK ACKNOWLEDGE
                                                                ;RECALIBRATE IF "SKI" OR "PIP" IS SET
                                                                ;VERIFY RECALIBRATION
040754  000404                          BR      1$              ;GO TO 1$ IF NO ERROR
040756  000240                          NOP                     ;RETURN HERE IF ERROR
040760  104000                          EMT                     ;ERROR # DEFINED BY TSTPRP SUBROUTINE
040762  000137  041256                  JMP     9$              ;GO TO 9$ IF ERROR
2272  040766                    1$:
2273  040766  012737  000000  001446    MOV     #0,RMDCO        ;CYLINDER = 0
2274  040774  012737  000000  001420    MOV     #0,RMDAO        ;TRACK = 0, SECTOR = 0
2275  041002  012737  010000  001444    MOV     #FMT16,RMOFO    ;16 BIT FORMAT
2276  041010  012737  000015  001412    MOV     #OFFSET!GO,RMCS1O       ;OFFSET COMMAND
2277  041016  012702  001545            MOV     #PUTINX,R2      ;SETUP REGISTER INDEX TABLE
2278  041022  112722  000006            MOVB    #RMDA,(R2)+     ;FOR SEEK TO CYLINDER 0
2279  041026  112722  000034            MOVB    #RMDC,(R2)+
```

```
                                                MOVB    #RMOF,(R2)+
                                                MOVB    #RMCS1,(R2)+
                                                MOVB    #200,(R2)+
                                                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                                BR      2$              ;GO TO 2$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        2$:
                                                JSR     PC,GETSTS       ;SETUP FOR STATUS FETCH
                                                JSR     PC,TIMOUT
                                        3$:
                                                MOVB    #RMCS1,PUTINX   ;LOAD REGISTER INDEX TABLE
                                                MOVB    #200,PUTINX+1
                                                MOV     #SEARCH!GO,RMCS10       ;STORE SEARCH COMMAND
                                                JSR     PC,PUT          ;GO WRITE REGISTER(S) WITH PUT SUBROUTINE
                                                BR      4$              ;GO TO 4$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PUT SUBROUTINE
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        4$:
                                                JSR     PC,TIMOUT       ;WAIT FOR SEARCH TO COMPLETE

                                                JSR     PC,GET          ;GO READ REGISTER(S) WITH GET SUBROUTINE
                                                BR      5$              ;GO TO 5$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY GET SUBROUTINE
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        5$:
                                                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                                BR      6$              ;GO TO 6$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        6$:
                                                JSR     PC,SCHSTS       ;GO VERIFY SEARCH OPERATION
                                                BR      7$              ;GO TO 7$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY SCHSTS SUBROUTINE
                                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        7$:
                                                JSR     PC,CMPERRSTS    ;CHECK ANY ERRORS NOT MASKED
                                                .WORD   NDTMSK          ;MASK FOR RMER1
                                                .WORD   DPE             ;MASK FOR RMER2
                                                BR      8$              ;GO TO 8$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY CMPERRSTS SUBROUTINE
                                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                                JMP     9$              ;GO TO 9$ IF ERROR
                                        8$:
                                                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                                BR      9$              ;GO TO 9$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                EMT                     ;ERROR # DEFINED BY SECERR SUBROUTINE
                                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

T67 SEARCH AT OFFSET

```
2304 041256                    9$:
```

END OF SUB-PASS ROUTINE

```
                                .SBTTL  END OF SUB-PASS ROUTINE

                                ;THIS IS THE END OF SUB-PASS ROUTINE. THIS ROUTINE IS USED TO
                                ;TERMINATE THE OPERATION OF THE CURRENT DEVICE UNDER TEST AND
                                ;SELECT THE NEXT DEVICE FOR TEST. IF THERE ARE NO MORE DEVICES
                                ;TO TEST, EXIT IS MADE TO '$EOP' ROUTINE. OTHERWISE, RETURN
                                ;IS MADE TO 'READY' ROUTINE.

041256  000004                  $EOSP:  SCOPE
041260  000240                          NOP
041262  013700  001466                  MOV     TSTQUE,R0       ;GET POINTER TO TSTQUE
041266  062700  000002                  ADD     #2,R0           ;ADJUST POINTER TO NEXT DEVICE
041272  010037  001466                  MOV     R0,TSTQUE       ;SAVE POINTER TO TSTQUE
041276  005710                          TST     (R0)            ;ANY MORE DEVICES FOR TEST ?
041300  001402                          BEQ     1$              ;BR IF NO
041302  000137  007574                  JMP     READY           ;YES, JUMP TO 'READY' ROUTINE
041306  012737  001470  001466  1$:     MOV     #TSTQUE+2,TSTQUE        ;INITIALIZE POINTER TO FIRST DEVICE IN
                                                                ;TEST QUE TABLE

                                .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

041314                          $EOP:
041314  000240                          NOP
041316  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
041322  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
041326  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
041332  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
041340  005327                          DEC     (PC)+           ;;LOOP?
041342  000001                  $EOPCT: .WORD   1
041344  003066                          BGT     $DOAGN          ;;YES
041346  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
041350  000001                  $ENDCT: .WORD   1
041352  041342                          $EOPCT
041354  104401  041362                  TYPE    ,65$            ;;TYPE ASCIZ STRING
041360  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
041400                          64$:
041400  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE PASS NUMBER
041404  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
041406  005737  001126                  TST     $ERTTL          ;;SEE IF ANY ERRORS THIS PASS
041412  001431                          BEQ     $GT42P          ;;BR IF NO ERRORS TO REPORT
041414  104401  041422                  TYPE    ,67$            ;;TYPE ASCIZ STRING
041420  000421                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
041464                          66$:
041464  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                ;;TOTAL NUMBER OF ERRORS
041470  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
041472  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
```

END OF PASS ROUTINE

```
041476  104401  001217          $GT42P: TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
041502  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
041506  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
041510  000005                          RESET                   ;;CLEAR THE WORLD
041512  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
041514  000240                          NOP                     ;;SAVE ROOM
041516  000240                          NOP                     ;;FOR
041520  000240                          NOP                     ;;ACT11
041522                          $DOAGN:
041522  000137                          JMP     @(PC)+          ;;RETURN
041524  007574                  $RTNAD: .WORD   READY
041526     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
                                        .EVEN
```

```
                        .SBTTL  TEST PREPARATION MODULE
                        ;
                        ; THIS MODULE PREPARES THE 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   NNNNNN          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 OR SKI ERROR
                        ;       BIT 10 = 1      RECALIBRATE DRIVE
                        ;       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
                        
 041532                 TSTPRP:
                        
                        ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
 041532  017637  000000  042456          MOV     @(SP),39$       ;STORE DESCRIPTOR
 041540  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 041544  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
 041550  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                        
 041554  004737  042462                  JSR     PC,GETSTS       ;SETUP TO READ ALL REGISTERS
 041560  004737  042546                  JSR     PC,GET          ;GET RMER2
 041564  000411                          BR      2$              ;BR IF NO ERROR DETECTED
 041566  000401                          BR      1$              ;GET OVER ERROR NUMBER
 041570  000000                          .WORD   0               ;ERROR DEFINED BY GET SUBROUTINE
 041572  062716  000004          1$:     ADD     #4,(SP)         ;XFER ERROR TO USER AND
 041576  113776  041570  000000          MOVB    1$-2,@(SP)      ;GET ERROR NUMBER.
 041604  000137  042446                  JMP     37$
                        
 041610  013737  001400  042460  2$:     MOV     RMER2I,40$      ;GET RMER2 AND SAVE FOR LATER
                        
                        ;;*****************************************************************
```

```
                                        ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
041616  005737  042456                        TST     39$             ;SELECT DEVICE??
041622  100014                              BPL     4$              ;NO!!

041624  004737  050774                      JSR     PC,DEVSEL       ;GO SELECT DEVICE
041630  000411                              BR      4$              ;NO ERROR - CONTINUE
041632  000401                              BR      3$
041634  000000                      3$:     .WORD   0               ;ERROR NUMBER FROM DEVSEL
041636  062716  000004                      ADD     #4,(SP)         ;TRANSFER ERROR TO USER
041642  113776  041634  000000            MOVB    3$-2,@(SP)
041650  000137  042446                      JMP     37$

                                        ;;**************************************************************************
                                        ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
041654                                  4$:
041654  032737  040000  042456              BIT     #BIT14,39$      ;CLEAR CONTROLLER??
041662  001451                              BEQ     13$             ;NO!!

041664  004737  052466                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
041670  000411                              BR      7$              ;CONTINUE - NO ERROR
041672  000401                              BR      6$
041674  000000                      5$:     .WORD   0               ;ERROR NUMBER FROM CNTCLR
041676  062716  000004              6$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
041702  113776  041674  000000            MOVB    5$,@(SP)
041710  000137  042446                      JMP     37$

                                        ;;**************************************************************************
                                        ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
041714                                  7$:
041714  032737  000100  042456              BIT     #BIT6,39$       ;VERIFY??
041722  001431                              BEQ     13$             ;NO!!

041724  004737  042546                      JSR     PC,GET          ;GO GET STATUS
041730  000411                              BR      10$             ;NO ERROR GETTING STATUS
041732  000401                              BR      9$
041734  000000                      8$:     .WORD   0               ;ERROR FROM GETTING STATUS
041736  062716  000004              9$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
041742  113776  041734  000000            MOVB    8$,@(SP)
041750  000137  042446                      JMP     37$

041754  004737  052604              10$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
041760  000412                              BR      13$             ;NO ERROR IN CLEAR
041762  000401                              BR      12$
041764  000000                      11$:    .WORD   0               ;ERROR IN STATUS CLEAR
041766  005726                      12$:    TST     (SP)+           ;STRIP RETURN ADDRESS TO
041770  062716  000004                      ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
041774  113776  041764  000000            MOVB    11$,@(SP)       ;ERROR TO USER
042002  000137  042446                      JMP     37$

                                        ;;**************************************************************************
                                        ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
                                        ;NOT VALID
042006                                  13$:
042006  032737  010000  042456              BIT     #BIT12,39$      ;PACK ACKNOWLEDGE??
042014  001501                              BEQ     25$             ;NO!!

042016  004737  042546                      JSR     PC,GET
```

```
                                        BR      16$             ;NO ERROR GETTING RMDS
                                        BR      15$
                                14$:    .WORD   0
                                15$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    14$,@(SP)
                                        JMP     37$

                                16$:    BIT     #VV,RMDSI       ;IS VOLUME VALID??
                                        BNE     25$             ;YES!!

                                        MOV     #PAKACK!GO,RMCS1O       ;LOAD PACK ACK COMMAND
                                        MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
                                        MOVB    #200,PUTINX+1
                                        JSR     PC,PUT          ;GO WRITE COMMAND
                                        BR      19$             ;NO ERROR LOADING REGISTER
                                        BR      18$
                                17$:    .WORD   0               ;ERROR FROM PUT SUB
                                18$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    17$,@(SP)
                                        BR      37$

                                19$:    JSR     PC,TIMOUT       ;WAIT FOR COMMAND TO COMPLETE

                                ;;**********************************************************************
                                ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
                                        BIT     #BIT4,39$       ;VERIFY PACK ACKNOWLEDGE??
                                        BEQ     25$             ;NO!!

                                        JSR     PC,GET          ;GO GET STATUS
                                        BR      22$             ;NO ERROR GETTING STATUS
                                        BR      21$
                                20$:    .WORD   0               ;ERROR FROM GET SUB
                                21$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    20$,@(SP)
                                        BR      37$

                                22$:    JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
                                        BR      25$             ;NO ERROR
                                        BR      24$
                                23$:    .WORD   0               ;PACK ACKNOWLEDGE ERROR
                                24$:    TST     (SP)+           ;STRIP RETURN TO SUB AND
                                        ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    23$,@(SP)
                                        BR      37$

                                ;;**********************************************************************
                                ;RECALIBRATE DRIVE IF BIT 11 IS SET IN TASK AND "SKI" IS SET
                                ;OR "PIP" IS ACTIVE.
                                ;RECALIBRATE DRIVE IF BIT 10 IS SET
                                25$:
                                        BIT     #BIT10,39$      ;RECALIBRATE DRIVE ?
                                        BNE     29$             ;YES!!
                                        BIT     #BIT11,39$      ;RECALIBRATE??
                                        BEQ     38$             ;NO!!

                                        JSR     PC,GET          ;GO GET RMDS
                                        BR      28$             ;NO ERROR GETTING RMDS
```

```
042246  000401                          BR      27$
042250  000000                  26$:    .WORD   0               ;ERROR FROM GET SUB
042252  062716  000004          27$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
042256  113776  042250  000000          MOVB    26$,@(SP)
042264  000470                          BR      37$

042266  032737  040000  042460  28$:    BIT     #SKI,40$        ;WAS SKI SET ?
042274  001004                          BNE     29$             ;YES, GO RECALIBRATE
042276  032737  020000  001350          BIT     #PIP,RMDS1      ;IS PIP ACTIVE??
042304  001462                          BEQ     38$             ;NO!!

042306  012737  000007  001412  29$:    MOV     #RECAL!GO,RMCS1O        ;LOAD RECALIBRATE COMMAND
042314  112737  000000  001545          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
042322  112737  000200  001546          MOVB    #200,PUTINX+1   ;SET TERMINATOR
042330  004737  043016                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
042334  000410                          BR      31$             ;NO ERROR
042336  000401                          BR      30$
042340  000000                          .WORD   0               ;ERROR IN REGISTER TRANSFER
042342  062716  000004          30$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
042346  113776  042340  000000          MOVB    30$-2,@(SP)
042354  000434                          BR      37$

042356  004737  043370          31$:    JSR     PC,TIMOUT       ;WAIT FOR COMPLETION

                                ;;*******************************************************************
                                ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
042362  032737  000010  042456          BIT     #BIT3,39$       ;VERIFY RECALIBRATE??
042370  001430                          BEQ     38$             ;NO!!

042372  004737  042546                  JSR     PC,GET          ;GO GET STATUS
042376  000410                          BR      34$             ;NO ERROR GETTING STATUS
042400  000401                          BR      33$
042402  000000                  32$:    .WORD   0               ;ERROR FROM GET
042404  062716  000004          33$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
042410  113776  042402  000000          MOVB    32$,@(SP)
042416  000413                          BR      37$

042420  004737  054260          34$:    JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
042424  000412                          BR      38$             ;NO ERROR DURING RECALIBRATE
042426  000401                          BR      36$
042430  000000                  35$:    .WORD   0               ;ERROR DURING RECALIBRATE
042432  005726                  36$:    TST     (SP)+           ;STRIP RETURN TO SUB AND
042434  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
042440  113776  042430  000000          MOVB    35$,@(SP)
042446  162716  000002          37$:    SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
042452  000240                  38$:    NOP
042454  000207                          RTS     PC              ;RETURN TO USER

042456  000000                  39$:    .WORD   0               ;TASK/VERIFY DESCRIPTOR
042460  000000                  40$:    .WORD   0               ;CONTAINS RMER2
```

GET STATUS SUBROUTINE

```
                                .SBTTL  GET STATUS SUBROUTINE
                                ;
                                ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
                                ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
                                ;AND THEN RETURNS TO THE USER.
                                ;
                                ;CALL:  JSR     PC,GETSTS
                                ;       ???                     RETURN HERE
                                ;
042462                          GETSTS:
042462  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042464  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042466  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042470  012700  001516                  MOV     #GETINX,R0      ;R0 = ADDRESS OF INDEX TABLE
042474  012701  001406                  MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
042500  012702  000046                  MOV     #RMEC2,R2       ;R2 = REGISTER INDE
042504  110220                  2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
042506  005041                          CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
042510  162702  000002          3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
042514  100405                          BMI     4$              ;BRANCH OUT IF DONE
042516  022702  000022                  CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
042522  001370                          BNE     2$
042524  005041                          CLR     -(R1)
042526  000770                          BR      3$
042530  112720  000200          4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
042534  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
042536  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
042540  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
042542  000240                          NOP
042544  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
                        ;       ???
                        ;
                        ;R0 = REGISTER BASE ADDRESS
                        ;R1 = REGISTER ADDRESS
                        ;R2 = BUFFER BASE ADDRESS
                        ;R3 = BUFFER ADDRESS
                        ;R4 = POINTER TO REGISTER INDEX

042546  000240          GET:    NOP
042550  062716  000004          ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
042554  105076  000000          CLRB    @(SP)           ;ERROR CALL
042560  162716  000004          SUB     #4,(SP)
042564  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042566  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042570  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042572  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
042574  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
042576  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
042602  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
042606  013700  001276          MOV     $BASE,R0
042612  012702  001336          MOV     #GETBUF,R2
042616  012704  001516          MOV     #GETINX,R4
042622  012737  042730  000004  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
042630  012737  000300  000006  MOV     #PR6,ERRVEC+2
042636  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
042644  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
042652  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
042660  001007                  BNE     3$              ;YES!!
042662  062766  000004  000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
042670  112776  000112  000016  MOVB    #112,@16(SP)    ;ERROR CALL
042676  000423                  BR      7$
042700  105714          3$:     TSTB    (R4)            ;DONE??
042702  100433                  BMI     9$              ;YES!!
042704  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
042706  042701  177700          BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
```

```
                                                ADD     R0,R1
                                                MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
                                                BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
                                                ADD     R2,R3
                                                MOV     (R1),(R3)       ;READ REGISTER
                                                BR      3$

                                        5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
                                                ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
                                                MOVB    #7,@16(SP)      ;USER'S ERROR CALL
                                        7$:     SUB     #2,16(SP)
                                        8$:     TSTB    (R4)            ;DONE CLEARING??
                                                BMI     9$              ;YES!!
                                                CLR     R3              ;CLEAR REMAINING STORAGE
                                                MOVB    (R4)+,R3        ;LOCATIONS
                                                ADD     R2,R3
                                                CLR     (R3)
                                                BR      8$
                                        9$:
042772  012637  000006                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
042776  012637  000004                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
043002  012604                                MOV     (SP)+,R4        ;;POP STACK INTO R4
043004  012603                                MOV     (SP)+,R3        ;;POP STACK INTO R3
043006  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
043010  012601                                MOV     (SP)+,R1        ;;POP STACK INTO R1
043012  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
043014  000207                                RTS     PC              ;RETURN
```

PUT SUBROUTINE

```
                                .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
                                ;       ???
                                ;
                                ;R0 = REGISTER BASE ADDRESS
                                ;R1 = REGISTER ADDRESS
                                ;R2 = BUFFER BASE ADDRESS
                                ;R3 = BUFFER ADDRESS
                                ;R4 = POINTER TO REGISTER INDEX

 043016  000240                PUT:    NOP
 043020  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 043022  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 043024  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 043026  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 043030  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 043032  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 043036  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
 043042  013700  001276                MOV     $BASE,R0
 043046  012702  001412                MOV     #PUTBUF,R2
 043052  012704  001545                MOV     #PUTINX,R4
 043056  012737  043176  000004        MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
 043064  012737  000300  000006        MOV     #PR6,ERRVEC+2
 043072  016037  000010  001174  1$:   MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
 043100  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
 043106  032737  004000  001176        BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
 043114  001007                        BNE     3$              ;YES!!
 043116  062766  000004  000016        ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
 043124  112776  000112  000016        MOVB    #112,@16(SP)    ;USER'S ERROR CALL
 043132  000430                        BR      7$
 043134  105714                3$:     TSTB    (R4)            ;DONE??
 043136  100431                        BMI     9$              ;YES!!
 043140  111401                        MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
 043142  042701  177700                BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
 043146  060001                        ADD     R0,R1
 043150  111403                        MOVB    (R4),R3         ;R3 = STORAGE ADDRESS
 043152  042703  177700                BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
 043156  060203                        ADD     R2,R3
 043160  011311                        MOV     (R3),(R1)       ;WRITE REGISTER
 043162  122714  000032                CMPB    #RMOF,(R4)      ;WAS RMOF JUST LOADED ?
```

```
043166  001001                          BNE     4$                      ;BR IF NO
043170  011311                          MOV     (R3),(R1)               ;WRITE RMOF AGAIN, TO ENSURE 16 BIT FORMAT
                                                                        ;IS SET BEFORE "SSEI" BIT IS SET.
043172  105724                  4$:     TSTB    (R4)+                   ;ADJUST REGISTER POINTER
043174  000757                          BR      3$

043176  022626                  5$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
043200  062766  000004  000016          ADD     #4,16(SP)               ;WRITE ERROR NUMBER IN
043206  112776  000007  000016          MOVB    #7,@16(SP)              ;USER'S ERROR CALL
043214  162766  000002  000016  7$:     SUB     #2,16(SP)

043222                          9$:
043222  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
043226  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
043232  012604                          MOV     (SP)+,R4                ;;POP STACK INTO R4
043234  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
043236  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
043240  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
043242  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
043244  000207                          RTS     PC                      ;RETURN
```

```
                                        .SBTTL  SIZE CLOCK SUBROUTINE

                                SIZCLK:
043246  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
043252  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
043256  012737  043314  000004          MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
043264  012737  000300  000006          MOV     #PR6,ERRVEC+2
043272  012737  177546  001512          MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
043300  012737  000100  001514          MOV     #100,CLKVCT
043306  005777  136200                  TST     @CLKADR         ;TEST FOR KW11-L PRESENT
043312  000421                          BR      3$              ;YES - KW11-L IS PRESENT
043314  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
043316  012737  043346  000004          MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
043324  012737  172540  001512          MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
043332  012737  000104  001514          MOV     #104,CLKVCT
043340  005777  136146                  TST     @CLKADR         ;TEST FOR KW11-P PRESENT
043344  000404                          BR      3$              ;YES - KW11-P IS PRESENT
043346  022626                  2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
043350  062766  000002  000004          ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
043356                          3$:
043356  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
043362  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
043366  000207                          RTS     PC              ;RETURN TO USER
```

```
                                .SBTTL  TIMEOUT SUBROUTINE
                                ;
                                ;THIS SUBROUTINE WAITS FOR RDY = 1 AND GO = 0 OR FOR A TIMEOUT
                                ;GREATER THAN APPROX. 500 MSEC AND THEN RETURNS.
                                ;
                                ;CALL:  JSR     PC,TIMOUT
                                ;       ???                             RETURN HERE
                                ;
043370                          TIMOUT:
043370  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043372  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043374  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
043400  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
043404  012737  043470  000004         MOV     #3$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
043412  012737  000300  000006         MOV     #PR6,ERRVEC+2
043420  013700  001276                 MOV     $BASE,R0        ;R0=BASE ADDRESS
043424  012702  000036                 MOV     #30.,R2         ;R2=NUMBER OF CLOCK CYCLES
043430  004737  043532         1$:     JSR     PC,STIMER       ;START CLOCK TIMER
                                ;
043434  016046  000000         2$:     MOV     RMCS1(R0),-(SP) ;GET STATUS
043440  042716  177576                 BIC     #^C<RDY!GO>,(SP)
043444  022726  000200                 CMP     #RDY,(SP)+      ;RDY=1,GO=0??
043450  001421                         BEQ     4$              ;YES!!
043452  032777  000200  136032         BIT     #BIT7,@CLKADR   ;TIMER DONE??
043460  001765                         BEQ     2$              ;NO!!
043462  005302                         DEC     R2              ;DEC NUMBER OF CYCLES
043464  001361                         BNE     1$              ;CONTINUE IF NOT DONE
043466  000412                         BR      4$              ;"RDY" DID NOT SET OR "GO" DID NOT RESET
                                                               ;WITHIN 500 MSEC AFTER THE COMMAND WAS ISSUED.
043470  022626                 3$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
043472  062766  000004  000012         ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
043500  112776  000007  000012         MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
043506  162766  000002  000012         SUB     #2,12(SP)
                                ;
043514                          4$:
043514  012637  000006                 MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
043520  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
043524  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
043526  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
043530  000207                         RTS     PC              ;RETURN TO USER
                                ;
                                ;THIS ROUTINE IS USED START THE KW11-L OR THE KW11-P CLOCK. THE CLOCK
                                ;IS STARTED TO RUN IN FLAG MODE. A CLOCK WILL BE 16.667MS FOR A 60 HZ CPU
                                ;AND 20MS FOR A 50 HZ CPU.
                                ;
043532                          STIMER:
043532  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043534  013701  001512                 MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
043540  020127  172540                 CMP     R1,#172540      ;KW11-P CLOCK??
043544  001003                         BNE     1$              ;NO!!
043546  012761  000001  000002         MOV     #1,2(R1)        ;SET COUNTER
043554  012711  000005         1$:     MOV     #BIT2!BIT0,(R1) ;START COUNTER
043560  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
043562  000207                         RTS     PC              ;RETURN
```

```
                                .SBTTL  PRIMARY ERROR CHECK SUBROUTINE

                                ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STATUS IS VALID AND
                                ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
                                ;FOLLOWING CHECKS ARE MADE:
                                ;
                                ;       .CORRECT 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

043564                          PRIERR:

                                ;CLEAR USER'S ERROR CALL
                                        ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
043564  062716  000004          CLRB    @(SP)                   ;CLEAR ERROR NUMBER
043570  105076  000000          SUB     #4,(SP)                 ;MOVE (SP) TO NO ERROR RETURN
043574  162716  000004

                                ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
043600  013737  001346  001142  MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
043606  042737  177770  001142  BIC     #^CUNTMSK,$BDDAT
043614  013737  001234  001140  MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
043622  042737  177770  001140  BIC     #^CUNTMSK,$GDDAT
043630  123737  001140  001142  CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
                                                        ;DRIVE NUMBERS
043636  001415                  BEQ     1$              ;YES!!
043640  062716  000004          ADD     #4,(SP)
043644  112776  000001  000000  MOVB    #1,@(SP)        ;ERROR 1
043652  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
043656  004736                  JSR     PC,@(SP)+       ;REPORT WRONG UNIT SELECTED
```

PRIMARY ERROR CHECK SUBROUTINE

```
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP
                                                JMP     10$             ;SKIP OTHER CHECKS
                                        1$:

                                        ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
                                        ;THE DEVICE IS NONEXISTANT
                                                BIT     #DVA,RMCS1I     ;DEVICE AVAILABLE??
                                                BNE     5$              ;YES!!
                                                MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
                                                BIS     #DVA,$GDDAT
                                                MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
                                                ADD     #4,(SP)
                                                MOVB    #2,@(SP)        ;ERROR #2
                                                BIT     #NED,RMCS2I     ;WAS NED SET??
                                                BEQ     2$              ;NO!!
                                                MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
                                                MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
                                                BIC     #NED,$GDDAT
                                                MOVB    #3,@(SP)        ;YES - CHANGE ERROR NUMBER
                                        2$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP
                                                BR      10$             ;SKIP OTHER CHECKS
                                        5$:

                                        ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
                                                BIT     #RDY,RMCS1I     ;CONTROLLER READY??
                                                BNE     7$              ;YES!!
                                                MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
                                                BIS     #RDY,$GDDAT
                                                BIC     #SC!TRE!MCPE!GO,$GDDAT
                                                MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
                                                ADD     #4,(SP)
                                                MOVB    #4,@(SP)        ;ERROR #4
                                                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT CONTROLLER NOT READY
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP
                                                BR      10$             ;SKIP OTHER CHECKS
                                        7$:

                                        ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
                                                BIT     #GO,RMCS1I      ;GO RESET??
                                                BEQ     8$              ;YES!!
                                                BIT     #DRY,RMDSI      ;DRIVE READY??
                                                BNE     8$              ;YES!!
                                                MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
                                                BIC     #SC!TRE!MCPE!GO,$GDDAT
                                                MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
                                                ADD     #4,(SP)
                                                MOVB    #5,@(SP)        ;ERROR #5
                                                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP
```

```
                                                BR      10$
                                        8$:

                                        ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
                                        ;PARITY ON THE MASSBUS CONTROL BUS
 044176  032737  020000  001336                 BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
 044204  001425                                 BEQ     9$              ;NO!!
 044206  013737  001336  001140                 MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
 044214  042737  160001  001140                 BIC     #SC!TRE!MCPE!GO,$GDDAT
 044222  013737  001336  001142                 MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
 044230  062716  000004                         ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
 044234  112776  000013  000000                 MOVB    #13,@(SP)       ;ERROR #47
 044242  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 044246  004736                                 JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 044250  162716  000010                         SUB     #10,(SP)        ;RESTORE STACK
 044254  000240                                 NOP
 044256  000453                                 BR      10$
 044260                                 9$:

                                        ;REPORT AN ERROR IF THE RM80 DETECTED A CONTROL BUS PARITY ERROR
 044260  032737  000010  001352                 BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
 044266  001451                                 BEQ     11$             ;NO!!
 044270  032737  000010  001400                 BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
 044276  001045                                 BNE     11$             ;NOT SURE!!
 044300  032737  000010  001426                 BIT     #PAR,RMER10     ;DID TEST SET PAR ??
 044306  001413                                 BEQ     93$             ;NO!!
 044310  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 044312  012700  001545                         MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
 044316  122710  000014                 91$:    CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
 044322  001002                                 BNE     92$
 044324  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
 044326  000431                                 BR      11$             ;PAR WAS SET BY TEST
 044330  105720                         92$:    TSTB    (R0)+           ;END OF TABLE??
 044332  100371                                 BPL     91$             ;NO!!
 044334  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
 044336  013737  001352  001140         93$:    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 044344  042737  000010  001140                 BIC     #PAR,$GDDAT
 044352  013737  001352  001142                 MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 044360  062716  000004                         ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
 044364  112776  000050  000000                 MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
 044372  162716  000002                         SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 044376  004736                                 JSR     PC,@(SP)+       ;REPORT THE ERROR
 044400  162716  000010                         SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 044404  000240                                 NOP
 044406  062716  000010                 10$:    ADD     #10,(SP)                ;RETURN TO ERROR
 044412  000240                         11$:    NOP                     ;RETURN TO NO ERROR
 044414  000207                                 RTS     PC
```

```
                                .SBTTL  SECONDARY ERROR CHECK SUBROUTINE

                                ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
                                ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER CONTENTS. THESE
                                ;ERRORS ARE DEEMED SECONDARY IN THAT THEY ARE NOT NECESSARILY
                                ;ASSOCIATED WITH THE OPERATION BEING PERFORMED.
                                ;WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES THE ERROR
                                ;NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
                                ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL. AFTER THE TEST ROUTINE
                                ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
                                ;OTHER ERRORS. WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
                                ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
                                ;
                                ;CALL:  JSR     PC,SECERR
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS

                                ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
                                ;INPUT REGISTER BUFFER.

044416                          SECERR:

                                ;;*************************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
044416  013737  001412  050556        MOV     RMCS10,515$     ;STORE FUNCTION CODE
044424  042737  177701  050556        BIC     #^C<F0!F1!F2!F3!F4>,515$
044432  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
044436  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
044442  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
044446  032737  000200  001350        BIT     #DRY,RMDSI      ;DRIVE READY??
044454  001024                        BNE     5$              ;YES!!
044456  013737  001350  001142        MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
044464  042737  177577  001142        BIC     #^CDRY,$BDDAT
044472  012737  000200  001140        MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
044500  062716  000004                ADD     #4,(SP)
044504  112776  000010  000000        MOVB    #10,@(SP)       ;ERROR NUMBER
044512  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
044516  004736                        JSR     PC,@(SP)+       ;REPORT NOT READY
044520  162716  000010                SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
044524  000240                        NOP

                                ;REPORT ERROR IF GO BIT IS NOT RESET
044526  032737  000001  001336  5$:   BIT     #GO,RMCS1I      ;GO BIT RESET??
044534  001423                        BEQ     10$             ;YES!!
044536  013737  001336  001142        MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
044544  042737  177776  001142        BIC     #^CGO,$BDDAT
044552  005037  001140                CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
044556  062716  000004                ADD     #4,(SP)
044562  112776  000011  000000        MOVB    #11,@(SP)       ;ERROR NUMBER
```

```
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP

                                ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
                                10$:    MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
                                        BIC     #^C76,$BDDAT
                                        MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
                                        CMP     $BDDAT,$GDDAT
                                        BEQ     15$             ;YES!!
                                        ADD     #4,(SP)
                                        MOVB    #12,@(SP)       ;ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP
                                15$:
                                ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
                                ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
                                ;OTHER ERRORS ARE SET
                                        CLR     $GDDAT          ;EXPECT "ERR" = 0
                                        TST     RMER1I          ;IS RMER1 = 0??
                                        BNE     20$             ;NO!!
                                        TST     RMER2I          ;IS RMERZ = 0??
                                        BEQ     25$             ;YES!!
                                20$:    BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
                                25$:    MOV     RMDSI,$BDDAT
                                        BIC     #^CERR,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
                                        BEQ     30$             ;YES!!
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
                                        MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                        JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
                                        SUB     #10,(SP)

                                ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
                                ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
                                ;SET TRE IS SET
                                30$:    CLR     $GDDAT          ;EXPECT "TRE" = 0
                                        MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
                                        BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
                                        BNE     35$             ;YES!!
                                        BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
                                        BEQ     40$             ;NO!!
                                        CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
                                        BHIS    40$             ;NO!!
                                35$:    BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
                                40$:    MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
                                        BIC     #^CTRE,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
                                        BEQ     45$             ;YES!!
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT TRE ERROR
```

```
    115 045072  162716  000010              SUB     #10,(SP)        ;RESTORE (SP)
    116 045076  000240                      NOP
    117 045100                          45$:
    118
    119                                 ;;*****************************************************************
    120                                 ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
    121                                 ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
    122                                 ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
    123                                 ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
    124                                 ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
    125                                 ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
    126
    127                                 ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
    128 045100  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
    129 045102  013700  050556              MOV     515$,R0         ;GET FUNCTION CODE
    130 045106  016037  070002  050550      MOV     FNCDTB(R0),500$ ;STORE ENTRY
    131 045114  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
    132
    133                                 ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
    134                                 ;ATA IS NOT SET AND SHOULD BE SET.
    135 045116  013737  050550  001140      MOV     500$,$GDDAT     ;GET EXPECTED ATA STATUS
    136 045124  032737  040000  001350      BIT     #ERR,RMDSI      ;IS COMPOSITE ERROR SET ??
    137 045132  001403                      BEQ     50$             ;NO !!
    138 045134  052737  100000  001140      BIS     #ATA,$GDDAT     ;EXPECT AN ATTENTION
    139 045142  042737  077777  001140  50$:BIC     #^CATA,$GDDAT   ;STRIP DONT CARES
    140 045150  013737  001350  001142      MOV     RMDSI,$BDDAT    ;GET RECEIVED ATA
    141 045156  042737  077777  001142      BIC     #^CATA,$BDDAT   ;STRIP DONT CARES
    142 045164  023737  001140  001142      CMP     $GDDAT,$BDDAT   ;IS ATA OK ??
    143 045172  001413                      BEQ     55$             ;YES !!
    144 045174  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
    145 045200  112776  000006  000000      MOVB    #6,@(SP)        ;LOAD ERROR # IN CALL
    146 045206  162716  000002              SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
    147 045212  004736                      JSR     PC,@(SP)+       ;REPORT ERROR
    148 045214  162716  000010              SUB     #10,(SP)        ;RESTORE SP
    149 045220  000240                      NOP
    150 045222                          55$:
    151
    152                                 ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
    153                                 ;WITH FUNCTION CODE TABLE
    154 045222  013737  050550  001140      MOV     500$,$GDDAT     ;GET EXPECTED ILF
    155 045230  042737  177776  001140      BIC     #^CILF,$GDDAT   ;CLEAR ALL OTHER BITS
    156 045236  013737  001352  001142      MOV     RMER1I,$BDDAT   ;GET RECEIVED ILF
    157 045244  042737  177776  001142      BIC     #^CILF,$BDDAT   ;CLEAR ALL OTHER BITS
    158 045252  023737  001140  001142      CMP     $GDDAT,$BDDAT   ;IS ILF OK ??
    159 045260  001412                      BEQ     60$             ;YES !!
    160 045262  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
    161 045266  112776  000254  000000      MOVB    #254,@(SP)      ;WRITE ERROR NUMBER IN CALL
    162 045274  162716  000002              SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
    163 045300  004736                      JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
    164 045302  162716  000010              SUB     #10,(SP)        ;MOVE SP TO NO ERROR
    165 045306  005037  001140          60$:CLR     $GDDAT          ;CLEAR EXPECTED STATUS
    166
    167                                 ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
    168 045312  013746  050550              MOV     500$,-(SP)      ;GET WCE STATUS ENABLE
    169 045316  052716  137777              BIS     #^CWCE,(SP)     ;SET ALL OTHER BITS
    170 045322  013737  001346  001142      MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
    171 045330  042637  001142              BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
```

```
045334  001412                          BEQ     90$             ;BRANCH IF WCE OK
045336  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
045342  112776  000026  000000          MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
045350  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
045354  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
045356  162716  000010                  SUB     #10,(SP)        ;RESTORE ERROR
045362                          90$:
                                ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
045362  013746  050550                  MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
045366  052716  157777                  BIS     #^COPI,(SP)     ;SET ALL OTHER BITS
045372  013737  001352  001142          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
045400  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
045404  001412                          BEQ     100$            ;BRANCH IF OPI OK
045406  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
045412  112776  000164  000000          MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
045420  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
045424  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
045426  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
045432                          100$:
                                ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
                                ;SET AND VV IS NOT RESET
045432  013746  050550                  MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
045436  032737  000100  001350          BIT     #VV,RMDSI       ;IS VV SET
045444  001402                          BEQ     105$            ;NO !!
045446  042716  010000                  BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
045452  052716  167777          105$:   BIS     #^CIVC,(SP)     ;SET ALL OTHER BITS
045456  013737  001400  001142          MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
045464  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
045470  001412                          BEQ     110$            ;BRANCH IF IVC OK
045472  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
045476  112776  000165  000000          MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
045504  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
045510  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
045512  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
045516                          110$:

                                ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                                ; ALL WRITE ERRORS, I.E.,
                                ;       RMER1 - WLE, WCF
                                ;       RMER2 - DPE
                                ;       RMCS2 - UPE.
                                ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
                                ;WRITE ERROR ENABLE BIT IS RESET.

                                ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
                                ;THE DRIVE IS NOT WRITE PROTECTED
045516  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
045522  032737  004000  050550          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
045530  001404                          BEQ     115$            ;NO !!
045532  032737  004000  001350          BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
045540  001002                          BNE     120$            ;YES !!
045542  042716  004000          115$:   BIC     #WLE,(SP)       ;RESET WLE ENABLE
045546  013737  001352  001142  120$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
045554  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WLE IF ENABLED
045560  001412                          BEQ     125$            ;BRANCH IF WLE OK
```

```
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                                MOVB    #23,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        125$:

                                        ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
                                                MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
                                                BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
                                                BNE     130$            ;YES !!
                                                BIC     #WCF,(SP)       ;DISABLE WCF ERROR
                                        130$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+,$BDDAT    ;RESET WCF IF ENABLED
                                                BEQ     135$            ;BRANCH IF WCF OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                                MOVB    #25,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR
                                                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        135$:

                                        ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
                                                MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
                                                BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
                                                BNE     140$            ;YES !!
                                                BIC     #DPE,(SP)       ;RESET DPE ENABLE
                                        140$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+,$BDDAT    ;RESET DPE IF ENABLED
                                                BEQ     145$            ;BRANCH IF DPE OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                                MOVB    #40,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR
                                                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        145$:

                                        ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
                                                MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
                                                BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
                                                BNE     150$            ;YES !!
                                                BIC     #UPE,(SP)       ;DISABLE UPE ERROR
                                        150$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+,$BDDAT    ;RESET UPE IF ENABLED
                                                BEQ     155$            ;BRANCH IF UPE OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                                MOVB    #24,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                        155$:

                                        ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
                                                MOV     500$,-(SP)      ;GET IAE ENABLE
                                                BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
                                                MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+, $BDDAT   ;CLEAR IAE IF ENABLED
```

```
046050  001412                          BEQ     160$            ;BRANCH IF IAE IS OK
046052  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046056  112776  000166  000000          MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
046064  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046070  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046072  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046076                          160$:

                                ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                                ; ALL READ/WRITE ERRORS, I.E.,
                                ;       RMCS1 - TRE
                                ;       RMCS2 - DLT,NEM,MXF
                                ;       RMDS  - LBT
                                ;       RMER1 - AOE
                                ;NOTE:
                                ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
                                ;CYLINDER REGISTER IS WRITTEN
                                ;NOTE:
                                ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
                                ;NOTE:
                                ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE

                                ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
046076  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
046102  032737  001000  050550          BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
046110  001002                          BNE     165$            ;YES !!
046112  042716  100000                  BIC     #DLT,(SP)       ;RESET DLT ENABLE
046116  013737  001346  001142  165$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
046124  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
046130  001412                          BEQ     170$            ;BRANCH IF DLT IS OK
046132  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046136  112776  000032  000000          MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
046144  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046150  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046152  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046156                          170$:

                                ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
046156  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
046162  032737  001000  050550          BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
046170  001002                          BNE     175$            ;YES !!
046172  042716  004000                  BIC     #NEM,(SP)       ;DISABLE NEM
046176  013737  001346  001142  175$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
046204  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
046210  001412                          BEQ     180$            ;BRANCH IF NEM IS OK
046212  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046216  112776  000167  000000          MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
046224  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046230  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046232  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046236                          180$:

                                ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
046236  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
046242  032737  001000  050550          BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
046250  001002                          BNE     185$            ;YES !!
```

```
                                BIC     #MXF,(SP)       ;DISABLE MXF ERROR
046256  013737  001346  001142  185$:  MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
046264  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
046270  001412                        BEQ     190$            ;BRANCH IF MXF IS OK
046272  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046276  112776  000033  000000        MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
046304  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046310  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046312  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046316                          190$:

                                ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
046316  012746  177777                MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
046322  032737  001000  050550        BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
046330  001404                        BEQ     191$            ;NO !!
046332  032737  002000  001350        BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
046340  001002                        BNE     195$            ;YES !!
046342  042716  001000          191$: BIC     #AOE,(SP)       ;DISABLE AOE
046346  013737  001352  001142  195$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
046354  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
046360  001412                        BEQ     200$            ;BRANCH IF AOE IS OK
046362  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046366  112776  000020  000000        MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
046374  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046400  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046402  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046406                          200$:

                                ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                                ;HEADER ERRORS, I.E.,
                                ;       RMER1 - HCRC,HCE,FER
                                ;       RMER2 - BSE,SSE

                                ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
046406  032737  002000  001370          BIT     #HCI,RMOFI      ;IS HCI SET ??
046414  001403                        BEQ     201$            ;NO !!
046416  042737  000200  050550        BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
046424                          201$:

                                ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
046424  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
046430  032737  000200  050550        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
046436  001002                        BNE     205$            ;YES !!
046440  042716  000400                BIC     #HCRC,(SP)      ;DISABLE HCRC
046444  013737  001352  001142  205$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
046452  042637  001142                BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
046456  001412                        BEQ     210$            ;BRANCH IF HCRC IS OK
046460  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
046464  112776  000035  000000        MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
046472  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
046476  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
046500  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
046504                          210$:

                                ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
046504  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
046510  032737  000200  050550        BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
```

```
                                        BNE     215$            ;YES !!
                                        BIC     #HCE,(SP)       ;DISABLE HCE
001142                          215$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
001142                                  BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
                                        BEQ     220$            ;BRANCH IF HCE IS OK
                                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
000000                                  MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                220$:
                                
                                ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
                                        MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
050550                                  BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
                                        BNE     225$            ;YES !!
                                        BIC     #FER,(SP)       ;DISABLE FER
001142                          225$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
001142                                  BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
                                        BEQ     230$            ;BRANCH IF FER OK
                                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
000000                                  MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                230$:
                                
                                ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
                                        MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
050550                                  BIT     #HCE,500$       ;ARE THEY ENABLED ??
                                        BNE     235$            ;YES !!
                                        BIC     #BSE,(SP)       ;DISABLE BSE
001142                          235$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
001142                                  BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
                                        BEQ     240$            ;BRANCH IF BSE OK
                                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
000000                                  MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                240$:
                                
                                ;RESET THE ENABLING BIT (HCE), IF SKIP SECTOR ERROR INHIBIT (SSEI)
                                ;SET OR FORMAT 16 (FMT) IS CLEAR.
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
050556                                  MOV     515$,R0         ;GET FUNCTION CODE
070002  050550                          MOV     FNCDTB(R0),500$ ;STORE FUNCTION TABLE ENTRY
                                        MOV     (SP)+,R0        ;;POP STACK INTO R0
010000  001370                          BIT     #FMT16,RMOFI    ;18 BIT FORMAT MODE ?
                                        BEQ     241$            ;BR IF YES
001000  001370                          BIT     #SSEI,RMOFI     ;IS SSEI SET ?
                                        BEQ     242$            ;BR IF NO
000200  050550                  241$:   BIC     #HCE,500$       ;DISABLE HEADER ERROR
                                242$:
                                
                                ;REPORT ERROR IF SSE IS SET AND HEADER ERRORS ARE NOT ENABLED
                                        MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
```

```
                                                BIT     #HCE,500$       ;ARE HCE'S VALID FOR THIS FUNCTION ?
                                                BNE     243$            ;YES !!
                                                BIC     #SSE,(SP)       ;DISABLE SSE
                                        001142  243$:   MOV     RMER2I,$BDDAT   ;GET RECIEVED STATUS
                                                BIC     (SP)+,$BDDAT    ;CLEAR SSE IF NOT VALID FOR FUNCTION
                                                BEQ     244$            ;BRANCH IF SSE OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                        000000  MOVB    #63,@(SP)       ;WRITE ERROR NUMBER
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                        244$:

                                        ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
                                        ;FIELD ERRORS, I.E.,
                                        ;       RMCS2 - MDPE
                                        ;       RMER1 - DCK,ECH
                                        ;NOTE:
                                        ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
                                        ;DCK IS SET.

                                        ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
                                                MOV     #-1,-(SP)       ;ASSUME ENABLED
                                                BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
                                                BNE     245$            ;YES !!
                                                BIC     #MDPE,(SP)      ;DISBALE MDPE
                                        001142  245$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
                                                BEQ     250$            ;BRANCH IF MDPE OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                        000000  MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                        250$:

                                        ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
                                                MOV     #-1,-(SP)       ;ASSUME ENABLED
                                                BIT     #ECH,500$       ;ARE THEY ENABLED ??
                                                BNE     255$            ;YES !!
                                                BIC     #DCK,(SP)       ;DISABLE DCK
                                        001142  255$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
                                                BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
                                                BEQ     260$            ;BRANCH IF DCK IS OK
                                                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                        000000  MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                        260$:

                                        ;REPORT ERROR IF ECH IS SET AND,
                                        ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
                                        ;       ECI IS SET, OR
                                        ;       DCK IS NOT SET.
                                                MOV     #-1,-(SP)       ;ASSUME ENABLED
                                                BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
```

```
514 047222 001410                          BEQ     265$            ;NO !!
515 047224 032737 004000 001370            BIT     #ECI,RMOFI      ;IS ECI SET ??
516 047232 001004                          BNE     265$            ;YES !!
517 047234 032737 100000 001352            BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
518 047242 001002                          BNE     270$            ;YES !!
519 047244 042716 000100           265$:   BIC     #ECH,(SP)       ;DISABLE ECH
520 047250 013737 001352 001142    270$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
521 047256 042637 001142                   BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
522 047262 001412                          BEQ     275$            ;BRANCH IF ECH IS OK
523 047264 062716 000004                   ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
524 047270 112776 000031 000000            MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
525 047276 162716 000002                   SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
526 047302 004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
527 047304 162716 000010                   SUB     #10,(SP)        ;MOVE SP TO NO ERROR
528 047310                         275$:
```

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

047310  022737  000030  050556        CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED ?
047316  103402                        BLO     280$            ;BR IF YES
047320  000137  050522                JMP     355$            ;NO - EXIT

                        ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
047324  013737  001340  001142 280$:  MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
047332  001421                        BEQ     285$            ;YES
047334  032737  040000  001336        BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
047342  001015                        BNE     285$            ;YES!!
047344  062716  000004                ADD     #4,(SP)
047350  112776  000015  000000        MOVB    #15,@(SP)       ;ERROR NUMBER
047356  005037  001140                CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
047362  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
047366  004736                        JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
047370  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
047374  000240                        NOP

                        ;REPORT ERROR IF RMBA IS NOT CORRECT
047376  013737  001340  001140 285$:  MOV     RMWCI,$GDDAT    ;GET WORD COUNT AT END OF TRANSFER AND
047404  163737  001414  001140        SUB     RMWCO,$GDDAT    ;SUBTRACT STARTING WORD COUNT.
047412  006337  001140                ASL     $GDDAT          ;* 2
047416  063737  001416  001140        ADD     RMBAO,$GDDAT    ;ADD STARTING BUS ADDRESS

047424  032737  000010  001346        BIT     #BAI,RMCS2I     ;WAS BUS ADDRESS INHIBIT (BAI) SET ??
047432  001403                        BEQ     290$            ;NO !!
047434  013737  001416  001140        MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED

047442  023737  001140  001342 290$:  CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
047450  001416                        BEQ     295$            ;YES!!
047452  013737  001342  001142        MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
047460  062716  000004                ADD     #4,(SP)
047464  112776  000016  000000        MOVB    #16,@(SP)       ;ERROR NUMBER
047472  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
047476  004736                        JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
047500  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
047504  000240                        NOP

                        ;COMPUTE NUMBER OF SECTORS TRANSFERRED FROM WORD COUNT
047506  005046          295$:  CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
047510  013746  001340        MOV     RMWCI,-(SP)     ;GET WORD COUNT AT END OF TRANSFER AND
047514  163716  001414        SUB     RMWCO,(SP)      ;SUBTRACT STARTING WORD COUNT.

047520  012746  000400        MOV     #256.,-(SP)     ;ASSUME 256. WORDS PER SECTOR
047524  032737  000002  001412 BIT     #BIT1,RMCS1O    ;HEADER & DATA COMMAND ??
047532  001402                BEQ     300$            ;NO !!
047534  062716  000002        ADD     #2,(SP)         ;CHANGE TO 258. WORDS PER SECTOR

047540  005266  000004 300$:  INC     4(SP)           ;INCREMENT SECTOR COUNT
047544  161666  000002        SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
047550  003373                BGT     300$            ;CONTINUE IF NOT DONE
047552  022626                CMP     (SP)+,(SP)+     ;RESTORE STACK

                        ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM
```

```
                                        ;NUMBER OF SECTORS
047554 013737 001446 050550             MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
047562 013737 001420 050552             MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
047570 013737 001420 050554             MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
047576 013737 001334 050560             MOV     LSTRK,520$      ;STORE LAST TRACK,
047604 000337 050560                    SWAB    520$            ;GET TRACK ADDRESS TO LO BYTE AND
047610 005237 050560                    INC     520$            ;INCREMENT TO GET TOTL # OF TRACKS.

047614 042737 000377 050552             BIC     #^C<TADMSK>,505$        ;SAVE TRACK ADDRESS BITS AND
047622 000337 050552                    SWAB    505$            ;SWAP TRACK ADDRESS TO LOW BYTE.
047626 042737 177400 050554             BIC     #^C<SADMSK>,510$        ;SAVE SECTOR ADDRESS BITS
047634 062637 050554                    ADD     (SP)+,510$

047640 032737 001000 001444             BIT     #SSEI,RMOFO     ;WAS SSEI SET ?
047646 001411                           BEQ     310$            ;NO !!

047650 023727 050554 000040             CMP     510$,#32.       ;SECTOR OVEFLOWED??
047656 103417                           BLO     315$            ;NO!!
047660 005237 050552                    INC     505$            ;INCREMENT TRACK
047664 162737 000040 050554             SUB     #32.,510$       ;ADJUST SECTOR

047672 023727 050554 000037     310$:   CMP     510$,#31.       ;SECTOR OVERFLOWED ?
047700 103406                           BLO     315$            ;NO !!
047702 005237 050552            314$:   INC     505$            ;INCREMENT TRACK
047706 162737 000037 050554             SUB     #31.,510$       ;ADJUST SECTOR
047714 000766                           BR      310$            ;TRY AGAIN

047716 023737 050552 050560     315$:   CMP     505$,520$       ;TRACK OVERFLOWED??
047724 103407                           BLO     320$            ;NO!!
047726 005237 050550                    INC     500$            ;INCREMENT CYLINDER
047732 163737 050560 050552             SUB     520$,505$       ;ADJUST TRACK
047740 000766                           BR      315$            ;TRY AGAIN
047742 000240                           NOP

                                        ;REPORT ERROR IF "SSEI" IS SET AFTER TRACK SWITCHING OCCURED
                                        ;THRU DATA TRANSFER
047744 005037 001140            320$:   CLR     $GDDAT          ;SETUP EXPECTED SSEI = 0
047750 013737 001370 001142             MOV     RMOFI,$BDDAT    ;GET RECIEVED DATA AND
047756 042737 176777 001142             BIC     #^CSSEI,$BDDAT  ;SAVE SSEI BIT FOR COMPARE.

047764 032737 001000 001444             BIT     #SSEI,RMOFO     ;WAS SSEI SET BEFORE DATA TRANSFER ?
047772 001420                           BEQ     322$            ;NO
047774 123737 001421 001345             CMPB    RMDAO+1,RMDAI+1 ;DID TRACK SWITCH OCCUR ?
050002 001411                           BEQ     321$            ;NO
050004 005737 001142                    TST     $BDDAT          ;IS SSEI STILL SET ?
050010 001430                           BEQ     324$            ;NO
050012 062716 000004                    ADD     #4,(SP)         ;POINT TO ERROR
050016 112776 000225 000000             MOVB    #225,@(SP)      ;DEFINE ERROR NUMBER
050024 000414                           BR      323$

                                        ;REPORT ERROR IF "SSEI" IS INCORRECT
050026 012737 001000 001140     321$:   MOV     #SSEI,$GDDAT    ;SETUP EXPECTED SSEI = 1
050034 023737 001140 001142     322$:   CMP     $GDDAT,$BDDAT   ;IS SSEI CORRECT ?
050042 001413                           BEQ     324$            ;YES
050044 062716 000004                    ADD     #4,(SP)         ;POINT TO ERROR
050050 112776 000226 000000             MOVB    #226,@(SP)      ;DEFINE ERROR NUMBER
050056 162716 000002            323$:   SUB     #2,(SP)         ;ADJUST RETURN TO ERROR
```

```
                                                JSR     PC,@(SP)+       ;REPORT ERROR
                                                SUB     #10.,(SP)       ;RESTORE (SP)
                                                NOP

                                        ;REPORT ERROR IF "LBT" IS NOT CORRECT
                                        324$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT = 0
                                                CMP     500$,#560.      ;SHOULD LBT BE SET??
                                                BLOS    325$            ;NO!!
                                                BIT     #IAE,RMER1I     ;WAS IAE SET ??
                                                BNE     325$            ;YES - LBT SHOULD NOT BE SET
                                                MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT = 1
                                        325$:   MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
                                                BIC     #^CLBT,$BDDAT
                                                CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
                                                BEQ     330$            ;YES!!
                                                ADD     #4,(SP)
                                                MOVB    #17,@(SP)       ;ERROR NUMBER
                                                SUB     #2.(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
                                                SUB     #10.,(SP)       ;RESTORE (SP)
                                                NOP

                                        ;REPORT ERROR IF "AOE" IS INCORRECT
                                        330$:   CLR     $GDDAT          ;SET FOR AOE = 0
                                                BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
                                                BNE     340$            ;YES-"AOE" SHOULD BE ZERO
                                                CMP     500$,#560.      ;SHOULD AOE BE SET??
                                                BLOS    340$            ;NO!!
                                                TST     505$            ;MAYBE
                                                BNE     335$            ;YES
                                                TST     510$
                                                BNE     335$            ;YES !!
                                                BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
                                                BEQ     340$            ;NO !!
                                                TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
                                                BEQ     340$            ;YES !!
                                        335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE = 1
                                                CLR     505$            ;CLEAR EXPECTED TRACK
                                                MOV     #1,510$         ;EXPECT SECTOR = 1
                                        340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
                                                BIC     #^CAOE,$BDDAT
                                                CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
                                                BEQ     345$            ;YES!!
                                                ADD     #4,(SP)
                                                MOVB    #20,@(SP)       ;ERROR NUMBER
                                                SUB     #2.(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
                                                SUB     #10.,(SP)       ;RESTORE (SP)
                                                NOP

                                        ;REPORT ERROR IF RMDA IS NOT CORRECT
                                        345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
                                                BNE     355$            ;YES - DONT CHECK RMDA,RMDC
                                                MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
                                                SWAB    $GDDAT
                                                MOVB    510$,$GDDAT
                                                MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
```

```
SECONDARY ERROR CHECK SUBROUTINE

 172 050404  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
 173 050412  001413                           BEQ     350$            ;BRANCH IF EQUAL
 174 050414  062716  000004                   ADD     #4,(SP)
 175 050420  112776  000021  000000           MOVB    #21,@(SP)       ;ERROR NUMBER
 176 050426  162716  000002                   SUB     #2.(SP)         ;MOVE SP TO RETURN FOR ERROR
 177 050432  004736                           JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
 178 050434  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
 179 050440  000240                           NOP
 180
 181                                  ;REPORT ERROR IF RMDC IS INCORRECT
 182 050442  013737  050550  001140  350$:    MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
 183 050450  042737  176000  001140           BIC     #^C1777,$GDDAT
 184 050456  013737  001372  001142           MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
 185 050464  023737  001140  001142           CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
 186 050472  001413                           BEQ     355$            ;BRANCH IF EQUAL
 187 050474  062716  000004                   ADD     #4,(SP)
 188 050500  112776  000022  000000           MOVB    #22,@(SP)       ;ERROR NUMBER
 189 050506  162716  000002                   SUB     #2.(SP)         ;MOVE SP TO RETURN FOR ERROR
 190 050512  004736                           JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
 191 050514  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
 192 050520  000240                           NOP
 193
 194 050522  062716  000004          355$:    ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
 195 050526  105776  000000                   TSTB    @(SP)           ;WAS ERROR FOUND??
 196 050532  001403                           BEQ     360$
 197 050534  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
 198 050540  000402                           BR      365$
 199 050542  162716  000004          360$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
 200 050546  000207          365$:    RTS     PC
 201
 202 050550  000000          500$:    .WORD   0               ;CYLINDER
 203 050552  000000          505$:    .WORD   0               ;TRACK
 204 050554  000000          510$:    .WORD   0               ;SECTOR
 205 050556  000000          515$:    .WORD   0               ;FUNCTION CODE
 206 050560  000000          520$:    .WORD   0               ;TOTAL # OF TRACKS = LAST TRACK +1
```

```
                                .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
                                ;
                                ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
                                ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
                                ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
                                ;THE MASKS ARE APPLIED.
                                ;
                                ;CALL:
                                ;(1)    JSR     PC,CMPERRSTS
                                ;       .WORD                   MASK FOR ERROR REGISTER 1
                                ;       .WORD                   MASK FOR ERROR REGISTER 2
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS
                                ;
                                ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
                                ;BE ZERO
                                ;
050562                          CMPERRSTS:
                                ;
                                ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
050562  013737  001352  001176         MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
050570  047637  000000  001176         BIC     @(SP),$TMP1     ;MASK   RMER1
050576  062716  000002                 ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
050602  013737  001400  001200         MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
050610  047637  000000  001200         BIC     @(SP),$TMP2     ;MASK RMER2
                                ;
                                ;
                                ;CLEAR USER'S ERROR CALL
050616  062716  000006                 ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
050622  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
050626  162716  000004                 SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
                                ;
                                ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
050632  005737  001176                 TST     $TMP1           ;ANY ERRORS TO REPORT??
050636  001420                         BEQ     5$              ;NO !!
050640  013737  001176  001142         MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
050646  005037  001140                 CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
050652  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
050656  112776  000066  000000         MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
050664  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
050670  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
050672  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
050676  000240                         NOP
050700                          5$:
                                ;
                                ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
050700  005737  001200                 TST     $TMP2           ;ANY ERRORS IN RMER2?
050704  001420                         BEQ     10$             ;NO!!
                                ;
050706  013737  001200  001142         MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
050714  005037  001140                 CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
050720  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
050724  112776  000067  000000         MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
```

```
050732  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
050736  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
050740  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
050744  000240                  NOP
050746                  10$:

                        ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
050746  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
050752  105776  000000          TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
050756  001403                  BEQ     20$             ;NO!!
050760  062716  000004          ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
050764  000402                  BR      30$
050766  162716  000004  20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
050772  000207          30$:    RTS     PC              ;RETURN TO USER
```

DEVICE SELECT SUBROUTINE

```
                                .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

050774                          DEVSEL:

                                ;CLEAR USER'S ERROR CALL
050774  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
051000  105076  000000                CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
051004  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK

                                ;SAVE USER'S INFORMATION AND SETUP REGISTERS
051010  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
051014  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
051020  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
051022  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
051024  012737  051144  000004        MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
051032  012737  000300  000006        MOV     #PR6,ERRVEC+2
051040  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
051044  013701  001466                MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER

                                ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
051050  111160  000010                MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
051054  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
051062  016037  000010  001174        MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS

051070  032737  010000  001174        BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT ?
051076  001407                        BEQ     10$             ;NO!!
051100  062766  000004  000010        ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
051106  112776  000111  000010        MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
051114  000422                        BR      30$

051116  032737  004000  001176  10$:  BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE ?
051124  001021                        BNE     35$             ;YES!!
051126  062766  000004  000010        ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
051134  112776  000112  000010        MOVB    #112,@10(SP)    ;WRITE ERROR NUMBER
051142  000407                        BR      30$

                                ;HANDLE BUS TIMEOUT
051144  022626                20$:    CMP     (SP)+,(SP)+     ;ADJUST SP
051146  062766  000004  000010        ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
051154  112776  000113  000010        MOVB    #113,@10(SP)    ;WRITE BUS TIMEOUT ERROR NUMBER
051162  162766  000002  000010  30$:  SUB     #2,10(SP)       ;ADJUST RETURN TO 'NOP' PRECEDING
                                                              ;THE ERROR CALL

                                ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
051170                          35$:
051170  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
051172  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
051174  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
```

DEVICE SELECT SUBROUTINE

```
051200  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 52 051204  000207                     RTS     PC              ;EXIT
```

```
                                .SBTTL  SEEK STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
                                ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
                                ;
                                ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
                                ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
                                ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
                                ;
                                ;CALL:
                                ;(1)     JSR     PC,SEKSTS
                                ;       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

051206                          SEKSTS:

                                ;CLEAR USERS' ERROR CALL
051206  000240                          NOP
051210  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
051214  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
051220  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
051224  005037  052464                  CLR     300$            ;CLEAR ERROR FLAGS

                                ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
                                ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
051230  032737  000010  001352          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
051236  001424                          BEQ     1$              ;NO!!
051240  032737  000010  001400          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
051246  001020                          BNE     1$              ;NOT SURE!!

                                ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
051250  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
051254  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
051262  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
051266  112776  000050  000000          MOVB    #50,@(SP)       ;ERROR #50
051274  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
051300  004736                          JSR     PC,@(SP)+
051302  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
051306  000447                          BR      3$              ;IAE SHOULD  BE ZERO

                                ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK, SECTOR, CYLINDER
                                ;AND THE STATUS OF "SSEI" BIT DURING A SEEK OPERATION. ALSO, SET "SKI"
                                ;IF CYLINDER ADDRESS IS TOO LARGE.
051310  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SETUP FOR IAE = 1
051316  052737  040000  052464          BIS     #SKI,300$       ;SETUP FOR SKI = 1
051324  023727  001446  001060          CMP     RMDC0,#560.     ;GREATER THAN LAST CYLINDER ?
051332  101035                          BHI     3$              ;YES - CYLINDER IS INVALID
051334  042737  040000  052464          BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG

051342  123737  001421  001335          CMPB    RMDA0+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
051350  101026                          BHI     3$              ;YES - TRACK IS INVALID

051352  123727  001420  000035          CMPB    RMDA0,#29.      ;SECTOR > 29. ?
```

```
     051360  101420                          BLOS    2$              ;BR IF NO
     051362  032737  010000  001444          BIT     #FMT16,RMOFO    ;18 BIT FORMAT ?
     051370  001416                          BEQ     3$              ;YES - SECTOR IS INVALID FOR 18 BIT MODE
     051372  123727  001420  000036          CMPB    RMDA0,#30.      ;SECTOR > 30. ?
     051400  101410                          BLOS    2$              ;BR IF NO
     051402  032737  001000  001444          BIT     #SSEI,RMOFO     ;IS SSEI CLEAR ?
     051410  001406                          BEQ     3$              ;YES - SECTOR IS INVALID FOR 16 BIT MODE
     051412  123727  001420  000037          CMPB    RMDA0,#31.      ;SECTOR > 31. ?
     051420  101002                          BHI     3$              ;YES - SECTOR IS INVALID
     051422  005037  001140          2$:     CLR     $GDDAT          ;"IAE" SHOULD = 0
                                     ;COMPARE EXPECTED AND RECIEVED "IAE" STATUS
     051426  013737  001352  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
     051434  042737  175777  001142          BIC     #^CIAE,$BDDAT   ;SAVE IAE BIT FOR COMPARE
     051442  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;CORRECT "IAE" STATUS ?
     051450  001004                          BNE     35$             ;BR IF NO
     051452  042737  040000  052464          BIC     #SKI,300$       ;CLEAR SKI FLAG
     051460  000413                          BR      5$              ;GO CHECK NEXT ERROR
     051462                          35$:
                                     ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
     051462  062716  000004                  ADD     #4,(SP)
     051466  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
     051474  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     051500  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
     051502  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
     051506  000240                          NOP
     051510                          5$:

                                     ;REPORT ANY IVC ERROR AS
                                     ;       IVC ERROR WITH VOLUME VALID ZERO
                                     ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
     051510  032737  010000  001400          BIT     #IVC,RMER2I     ;IVC ERROR??
     051516  001427                          BEQ     52$             ;NO!!
     051520  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
     051524  013737  001400  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
     051532  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
     051536  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
     051544  032737  000100  001350          BIT     #VV,RMDSI
     051552  001403                          BEQ     51$
     051554  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
     051562  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     051566  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
     051570  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
     051574  000240                          NOP

     051576  013737  001400  001142  52$:    MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
     051604  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
     051612  013737  052464  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
     051620  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
     051626  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED

                                     ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
     051630  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
     051636  001032                          BNE     54$             ;YES !!
     051640  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
     051644  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
```

```
                                                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                                BR      6$              ;GO TO NEXT ERROR CHECK
                                        53$:

                                        ;REPORT ERROR IF SKI IS SET
                                                BIT     #SKI,$BDDAT     ;IS SKI SET ??
                                                BEQ     54$             ;NO - SKI IS OK
                                                ADD     #4,(SP)         ;MOVE (SP) TO ERROR
                                                MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
                                                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT SEEK ERROR
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP

                                        ;REPORT ANY DEVICE CHECK
                                        54$:    BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
                                                BEQ     6$              ;NO!!
                                                CLR     $GDDAT          ;EXPECTED STATUS
                                                MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
                                                ADD     #4,(SP)
                                                MOVB    #55,@(SP)       ;ERROR  #55
                                                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
                                                SUB     #10,(SP)        ;RESTORE SP
                                                NOP

                                        ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  = 0, OR  OPI
                                        ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
                                        6$:     BIT     #OPI,RMER1I     ;"OPI" ERROR??
                                                BEQ     8$              ;NO!!
                                                CLR     $GDDAT          ;EXPECTED STATUS
                                                MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
                                                ADD     #4,(SP)         ;MOVE (SP) TO ERROR
                                                MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
                                                BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
                                                BEQ     7$              ;NO!!
                                                MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
                                        7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                                JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
                                                SUB     #10,(SP)        ;RESTORE (SP)
                                                NOP

                                        ;SEE IF "PIP" = 0, AND "ATA", "MOL" AND "VV" = 1
                                        8$:     MOV     RMDSI,-(SP)
                                                BIC     #^C<ATA!PIP!MOL!VV>,(SP)
                                                CMP     #ATA!MOL!VV,(SP)+
                                                BNE     9$              ;ERROR IN RMDS
                                                JMP     14$             ;RMDS IS OK

                                        ;REPORT ERROR IF MOL = 0 AND OPI = 0
                                        9$:     BIT     #MOL,RMDSI      ;IS MOL RESET??
                                                BNE     10$             ;NO - MOL IS SET
                                                BIT     #OPI,RMER1I     ;WAS OPI SET
                                                BNE     10$             ;YES - DONT REPORT ERROR
                                                MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
```

```
                                        BIS     #MOL,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)
                                        MOVB    #62,@(SP)
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
                                        SUB     #10,(SP)
                                        NOP

                                ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
                                10$:    BIT     #PIP,RMDSI      ;IS "PIP" STILL SET??
                                        BEQ     11$             ;NO!!
                                        BIT     #SKI,RMER2I     ;WAS "SKI"SET??
                                        BNE     11$             ;YES-DONT REPORT PIP
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIC     #PIP,$BDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE (SP) TO ERROR
                                        MOVB    #56,@(SP)       ;LOAD ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT "PIP" SET AFTER SEEK
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP

                                ;REPORT AN ERROR IF "ATA" IS NOT SET
                                11$:    BIT     #ATA,RMDSI      ;WAS "ATA" SET ??
                                        BNE     13$             ;YES!!
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIS     #ATA!MOL!DPR!DRY,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE (SP) TO ERROR
                                        MOVB    #57,@(SP)       ;LOAD ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT ATTENTION NOT SET DURING
                                                                ;SEEK TEST
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP

                                ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
                                13$:    BIT     #VV,RMDSI       ;IS VV = 0 ??
                                        BNE     14$             ;NO!!
                                        BIT     #IVC,RMER2I     ;IS IVC ALSO 0 ??
                                        BNE     14$             ;NO - IVC IS SET
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIS     #VV,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)
                                        MOVB    #64,@(SP)       ;ERROR #64
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+
                                        SUB     #10,(SP)
                                        NOP
                                14$:

                                ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
                                        NOP
                                        ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
```

SEEK STATUS CHECK SUBROUTINE

```
    229 052442  105776  000000                TSTB    @(SP)           ;WAS ERROR CALLED??
    230 052446  001403                        BEQ     15$             ;NO!!
    231 052450  062716  000004                ADD     #4,(SP)         ;MOVE TO ERROR RETURN
    232 052454  000402                        BR      16$
    233
    234 052456  162716  000004    15$:        SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
    235 052462  000207            16$:        RTS     PC              ;RETURN
    236
    237 052464  000000            300$:       .WORD   0               ;ERROR FLAGS
```

```
                                .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
                                ;       ???

052466                          CNTCLR:
052466  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
052470  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
052472  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
052476  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
052502  012737  052542  000004          MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
052510  012737  000300  000006          MOV     #PR6,ERRVEC+2
052516  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
052522  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
052530  013701  001466                  MOV     TSTQUE,R1       ;GET DEVICE UNDER TEST
052534  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
052540  000412                          BR      20$

052542  022626                  10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
052544  062766  000004  000010          ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
052552  112776  000007  000010          MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
052560  162766  000002  000010          SUB     #2,10(SP)       ;ADJUST SP TO RETURN TO ERROR
052566                          20$:
052566  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
052572  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
052576  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
052600  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
052602  000207                          RTS     PC
```

```
                                .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE VERIFIES THAT THE SUBSYSTEM IS INITIALIZED BASED ON
                                ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
                                ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
                                ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.

                                ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
                                ;FOLLOWING STATUS BITS ARE NOT CHECKED:
                                ;
                                ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
                                ;

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

052604                          CLRSTS:

                                ;CLEAR USER'S ERROR CALL
052604  062716  000004                ADD     #4,(SP)                 ;MOVE SP TO ERROR
052610  105076  000000                CLRB    @(SP)                   ;CLEAR ERROR NUMBER
052614  162716  000004                SUB     #4,(SP)                 ;MOVE SP BACK TO NO ERROR
                                ;REPORT ERROR IF RMCS1 NOT INITIALIZED
052620  013737  001336  001142 4$:     MOV     RMCS1I,$BDDAT           ;VERIFY RMCS1
052626  042737  100000  001142        BIC     #SC,$BDDAT              ;IGNORE SPECIAL CONDITION
052634  012737  004200  001140        MOV     #DVA!RDY,$GDDAT         ;EXPECT DVA & RDY
052642  023737  001140  001142        CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED, RECEIVED
052650  001413                      BEQ     5$                      ;BRANCH IF EQUAL
052652  062716  000004                ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
052656  112776  000126  000000        MOVB    #126,@(SP)              ;WITE ERROR NUMBER IN CALL
052664  162716  000002                SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
052670  004736                      JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
052672  162716  000010                SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
052676  000240                      NOP
                                ;REPORT ERROR IF RMBA NOT RESET
052700  005037  001140 5$:     CLR     $GDDAT                  ;VERIFY RMBA IS ZERO
052704  013737  001342  001142        MOV     RMBAI,$BDDAT
052712  001413                      BEQ     7$                      ;BRANCH IF ZERO
052714  062716  000004                ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
052720  112776  000127  000000        MOVB    #127,@(SP)              ;WITE ERROR NUMBER IN CALL
052726  162716  000002                SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
052732  004736                      JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
052734  162716  000010                SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
052740  000240                      NOP
                                ;REPORT ERROR IF RMCS2 NOT INITIALIZED
052742  013737  001346  001142 7$:     MOV     RMCS2I,$BDDAT           ;VERIFY RMCS2
052750  010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
052752  005046                      CLR     -(SP)                   ;EXPECT IR & UNIT NUMBER
052754  013701  001466                MOV     TSTQUE,R1               ;R1 = ADDRESS OF TEST QUE
052760  111116                      MOVB    (R1),(SP)
052762  052716  000100                BIS     #IR,(SP)
052766  012637  001140                MOV     (SP)+,$GDDAT
```

CONTROLLER CLEAR STATUS CHECK SUBROUTINE

```
 58 052772  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
 59 052774  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
 60 053002  001413                          BEQ     9$                  ;BRANCH IF EQUAL
 61 053004  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
 62 053010  112776  000130  000000          MOVB    #130,@(SP)          ;WITE ERROR NUMBER IN CALL
 63 053016  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
 64 053022  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
 65 053024  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
 66 053030  000240                          NOP
 67                                 ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
 68 053032  005037  001140          9$:     CLR     $GDDAT              ;VERIFY RMER1
 69 053036  013737  001352  001142          MOV     RMER1I,$BDDAT
 70 053044  042737  040000  001142          BIC     #UNS,$BDDAT         ;IGNORE UNSAFE
 71 053052  001413                          BEQ     13$                 ;BRANCH IF ZERO
 72 053054  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
 73 053060  112776  000131  000000          MOVB    #131,@(SP)          ;WITE ERROR NUMBER IN CALL
 74 053066  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
 75 053072  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
 76 053074  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
 77 053100  000240                          NOP
 78                                 ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
 79 053102  013737  001362  001142  13$:     MOV     RMMR1I,$BDDAT       ;VERIFY RMMR
 80 053110  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT           ;IGNORE WORD CLOCK, SCT, TRK
 81 053116  012737  000010  001140          MOV     #MWD,$GDDAT         ;EXPECT WRITE DATA BIT
 82 053124  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED AND RECEIVED
 83 053132  001413                          BEQ     17$                 ;BRANCH IF 0
 84 053134  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
 85 053140  112776  000133  000000          MOVB    #133,@(SP)          ;WITE ERROR NUMBER IN CALL
 86 053146  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
 87 053152  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
 88 053154  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
 89 053160  000240                          NOP
 90                                 ;REPORT AN ERROR IF RMEC2 IS NOT RESET
 91 053162  005037  001140          17$:    CLR     $GDDAT              ;EXPECT ZEROS
 92 053166  013737  001404  001142          MOV     RMEC2I,$BDDAT       ;VERIFY RMEC2 = 0
 93 053174  001413                          BEQ     19$
 94 053176  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
 95 053202  112776  000135  000000          MOVB    #135,@(SP)          ;WITE ERROR NUMBER IN CALL
 96 053210  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
 97 053214  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
 98 053216  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
 99 053222  000240                          NOP
100                                 ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
101 053224  013737  001376  001142  19$:     MOV     RMMR2I,$BDDAT       ;VERIFY RMMR2
102 053232  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
103 053240  012737  011777  001140          MOV     #TST!1777,$GDDAT            ;EXPECT TEST BIT ON
104 053246  023737  001140  001142          CMP     $GDDAT,$BDDAT
105 053254  001413                          BEQ     21$
106 053256  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
107 053262  112776  000136  000000          MOVB    #136,@(SP)          ;WITE ERROR NUMBER IN CALL
108 053270  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
109 053274  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
110 053276  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
111 053302  000240                          NOP
112                                 ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
113 053304  005037  001140          21$:    CLR     $GDDAT              ;EXPECT ALL ZEROS
114 053310  013737  001400  001142          MOV     RMER2I,$BDDAT       ;VERIFY RMER2
```

CONTROLLER CLEAR STATUS CHECK SUBROUTINE

```
115 053316  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
116 053324  001413                          BEQ     215$            ;BRANCH IF OTHER BITS 0
117 053326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
118 053332  112776  000174  000000          MOVB    #174,@(SP)      ;WITE ERROR NUMBER IN CALL
119 053340  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
120 053344  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
121 053346  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
122 053352  000240                          NOP
123                                 ;REPORT ERROR IF RMDS NOT INITIALIZED
124 053354  013737  001350  001142  215$:   MOV     RMDSI,$BDDAT    ;TEST DRIVE STATUS REGISTER
125 053362  042737  177177  001142          BIC     #^C<DRY!DPR>,$BDDAT
126 053370  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
127 053376  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
128 053404  001413                          BEQ     22$             ;BRANCH IF EQUAL
129 053406  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
130 053412  112776  000134  000000          MOVB    #134,@(SP)      ;WRITE ERROR NUMBER
131 053420  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
132 053424  004736                          JSR     PC,@(SP)+       ;REPORT ERROR TO USER
133 053426  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
134 053432  000240                          NOP
135 053434  062716  000004          22$:    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
136 053440  105776  000000                  TSTB    @(SP)           ;WAS AN ERROE DETECTED??
137 053444  001403                          BEQ     23$             ;NO!!
138 053446  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
139 053452  000402                          BR      24$
140 053454  162716  000004          23$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
141 053460  000240                  24$:    NOP
142 053462  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

053464                          ACKSTS:

                                ;CLEAR USER'S ERROR CALL
053464  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053470  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
053474  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK

                                ;REPORT AN ERROR  IF "VV" IS 0
053500  032737  000100  001350          BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
053506  001024                          BNE     1$              ;YES!!
053510  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
053516  052737  000100  001140          BIS     #VV,$GDDAT
053524  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
053532  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053536  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
053544  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053550  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
053552  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
053556  000240                          NOP
053560                          1$:

                                ;REPORT AN ERROR IF "MOL" IS 0
053560  032737  010000  001350          BIT     #MOL,RMDSI      ;IS MOL SET??
053566  001024                          BNE     2$              ;YES!!
053570  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
053576  052737  010000  001140          BIS     #MOL,$GDDAT
053604  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
053612  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053616  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
053624  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053630  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
053632  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
053636  000240                          NOP
053640                          2$:

                                ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
053640  032737  060007  001352          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
053646  001570                          BEQ     7$

                                ;REPORT AN ERROR IF "UNS" IS SET
053650  032737  040000  001352          BIT     #UNS,RMER1I     ;WAS UNS SET??
053656  001424                          BEQ     3$              ;NO!!
053660  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

PACK ACKNOWLEDGE STATUS CHECK

```
 58 053666  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 59 053674  042737  040000  001140          BIC     #UNS,$GDDAT
 60 053702  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 61 053706  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 62 053714  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 63 053720  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 64 053722  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 65 053726  000240                          NOP
 66 053730                          3$:
 67
 68                                 ;REPORT ANY OPI ERROR
 69 053730  032737  020000  001352          BIT     #OPI,RMER1I     ;WAS OPI SET ??
 70 053736  001424                          BEQ     4$              ;NO!!
 71 053740  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 72 053746  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 73 053754  042737  020000  001140          BIC     #OPI,$GDDAT
 74 053762  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 75 053766  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 76 053774  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 77 054000  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 78 054002  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 79 054006  000240                          NOP
 80 054010                          4$:
 81
 82                                 ;REPORT ANY RMR ERROR
 83 054010  032737  000004  001352          BIT     #RMR,RMER1I     ;WAS RMR SET??
 84 054016  001424                          BEQ     5$              ;NO!!
 85 054020  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 86 054026  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
 87 054034  042737  000004  001140          BIC     #RMR,$GDDAT
 88 054042  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 89 054046  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
 90 054054  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 91 054060  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
 92 054062  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
 93 054066  000240                          NOP
 94 054070                          5$:
 95
 96                                 ;REPORT ANY     ILR ERROR
 97 054070  032737  000002  001352          BIT     #ILR,RMER1I     ;WAS ILR SET??
 98 054076  001424                          BEQ     6$              ;NO!!
 99 054100  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
100 054106  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
101 054114  042737  000002  001140          BIC     #ILR,$GDDAT
102 054122  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
103 054126  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
104 054134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
105 054140  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
106 054142  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
107 054146  000240                          NOP
108 054150                          6$:
109
110                                ;REPORT ANY     ILF ERROR
111 054150  032737  000001  001352          BIT     #ILF,RMER1I     ;WAS ILF SET??
112 054156  001424                          BEQ     7$              ;NO!!
113 054160  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
114 054166  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
```

PACK ACKNOWLEDGE STATUS CHECK

```
    115 054174  042737  000001  001140          BIC     #ILF,$GDDAT
    116 054202  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
    117 054206  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
    118 054214  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
    119 054220  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
    120 054222  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
    121 054226  000240                          NOP
    122 054230                          7$:
    123
    124                                 ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
    125 054230  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
    126 054234  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
    127 054240  001403                          BEQ     8$              ;NO!!
    128 054242  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
    129 054246  000402                          BR      9$
    130 054250  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
    131 054254  000240                  9$:     NOP
    132 054256  000207                          RTS     PC
```

```
                        .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE

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

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

054260                  RCLSTS:

                        ;CLEAR USER'S ERROR NUMBER
054260  062716  000004          ADD     #4,(SP)
054264  105076  000000          CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
054270  162716  000004          SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH

                        ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
054274  032737  022011  001352  BIT     #OPI!PAR!ILF!IAE,RMER1I
054302  001553                  BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK

                        ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
                        ;"PAR" = 1 AND "DPE" = 0
054304  032737  000010  001352  BIT     #PAR,RMER1I     ;WAS "PAR" SET??
054312  001430                  BEQ     1$              ;NO!!
054314  032737  000010  001400  BIT     #DPE,RMER2I     ;WAS "DPE" SET??
054322  001024                  BNE     1$              ;YES - NOT A REGISTER ERROR
054324  013737  001352  001140  MOV RMER1I,$GDDAT       ;EXPECTED STATUS
054332  042737  000010  001140  BIC     #PAR,$GDDAT
054340  013737  001352  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054346  062716  000004          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
054352  112776  000050  000000  MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
054360  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054364  004736                  JSR     PC,@(SP)+       ;GO REPORT ERROR
054366  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054372  000240                  NOP
054374                  1$:

                        ;REPORT ANY "ILF" ERROR
054374  032737  000001  001352  BIT     #ILF,RMER1I     ;WAS "ILF" SET??
054402  001424                  BEQ     2$              ;NO!!
054404  013737  001352  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
054412  042737  000001  001140  BIC     #ILF,$GDDAT
054420  013737  001352  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054426  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
054432  112776  000071  000000  MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
054440  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054444  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
054446  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054452  000240                  NOP
054454                  2$:
```

```
                                        ;REPORT ANY "OPI" ERROR AS
                                        ;       . OPI DUE TO "MOL" = 0
                                        ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
054454  032737  020000  001352                BIT     #OPI,RMER1I     ;WAS OPI SET??
054462  001433                            BEQ     31$             ;NO!!
054464  013737  001352  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
054472  042737  020000  001140            BIC     #OPI,$GDDAT
054500  013737  001352  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054506  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
054512  112776  000072  000000            MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
054520  032737  010000  001350            BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
054526  001403                            BEQ     3$              ;YES!!
054530  112776  000073  000000            MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
054536  162716  000002          3$:       SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054542  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
054544  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054550  000240                            NOP
054552                                  31$:

                                        ;REPORT AN ERROR IF "IAE" IS SET
054552  032737  002000  001352            BIT     #IAE,RMER1I     ;IS "IAE" SET??
054560  001424                            BEQ     4$              ;NO!!
054562  013737  001352  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
054570  042737  002000  001140            BIC     #IAE,$GDDAT
054576  013737  001352  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054604  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
054610  112776  000070  000000            MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
054616  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054622  004736                            JSR     PC,@(SP)+       ;REPORT ERROR
054624  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
054630  000240                            NOP
054632                                  4$:

                                        ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
054632  032737  050200  001400            BIT     #SKI!IVC!DVC,RMER2I
054640  001517                            BEQ     8$              ;NONE OF THE BITS ARE SET


                                        ;REPORT ANY "IVC" ERROR AS
                                        ;       .IVC WITH VV = 0
                                        ;       .ERRONEOUS IVC ERROR
054642  032737  010000  001400            BIT     #IVC,RMER2I     ;WAS IVC SET??
054650  001433                            BEQ     6$              ;NO!!
054652  013737  001400  001140            MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
054660  042737  010000  001140            BIC     #IVC,$GDDAT
054666  013737  001400  001142            MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
054674  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
054700  112776  000074  000000            MOVB    #74,@(SP)       ;WRITE ERROR NUMBER IN CALL
054706  032737  000100  001350            BIT     #VV,RMDSI       ;WAS VV = 0??
054714  001403                            BEQ     5$              ;YES!!
054716  112776  000075  000000            MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
054724  162716  000002          5$:       SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054730  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
054732  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054736  000240                            NOP
054740                                  6$:
```

```
                                        ;REPORT ANY "SKI" ERROR
054740  032737  040000  001400                BIT     #SKI,RMER2I     ;WAS SKI SET??
054746  001424                              BEQ     7$              ;NO!!
054750  013737  001400  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
054756  042737  040000  001140              BIC     #SKI,$GDDAT
054764  013737  001400  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
054772  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
054776  112776  000076  000000              MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
055004  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055010  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
055012  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO BRANCH
055016  000240                              NOP
055020                                  7$:

                                        ;REPORT ANY "DVC" ERROR
055020  032737  000200  001400                BIT     #DVC,RMER2I     ;WAS "DVC" SET??
055026  001424                              BEQ     8$              ;NO!!
055030  013737  001400  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
055036  042737  000200  001140              BIC     #DVC,$GDDAT
055044  013737  001400  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
055052  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
055056  112776  000077  000000              MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
055064  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055070  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
055072  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
055076  000240                              NOP
055100                                  8$:

                                        ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
055100  013746  001350                      MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
055104  042716  047676                      BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
055110  022726  110100                      CMP     #ATA!MOL!VV,(SP)+
055114  001002                              BNE     85$
055116  000137  055532                      JMP     13$
055122                                  85$:

                                        ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
                                        ;LINE AFTER RECALIBRATE WAS INITIATED
055122  032737  010000  001350                BIT     #MOL,RMDSI      ;DID MOL DROP??
055130  001030                              BNE     9$              ;NO!!
055132  032737  020000  001352              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
055140  001024                              BNE     9$              ;YES - DON'T REPORT MOL = 0
055142  013737  001350  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
055150  052737  010000  001140              BIS     #MOL,$GDDAT
055156  013737  001350  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
055164  062716  000004                      ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
055170  112776  000100  000000              MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
055176  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055202  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
055204  162716  000010                      SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
055210  000240                              NOP
055212                                  9$:

                                        ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
055212  032737  000100  001350                BIT     #VV,RMDSI       ;DID "VV" DROP??
055220  001030                              BNE     10$             ;NO!!
055222  032737  010000  001400              BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
```

```
                                        BNE     10$             ;YES - DONT REPORT VV = 0
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        BIS     #VV,$GDDAT
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+
                                        SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
                                        NOP
                                10$:

                                ;REPORT AN ERROR IF ATA IS NOT SET
                                        BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
                                        BNE     11$             ;YES!!
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIS     #ATA,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)
                                        JSR     PC,@(SP)+
                                        SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
                                        NOP

                                11$:

                                ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
                                ;ALWAYS CLEAR OFFSET MODE
                                        BIT     #OM,RMDSI       ;WAS "OM" RESET??
                                        BEQ     12$             ;YES!!
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIC     #OM,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
                                        SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
                                        NOP
                                12$:

                                ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
                                ;CYLINDER
                                        BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
                                        BEQ     13$             ;NO!!
                                        BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
                                        BNE     13$             ;YES-DONT REPORT "PIP"
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIC     #PIP,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+
                                        SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
                                        NOP
```

```
                                13$:

                                ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
055532  032737  040006  001352            BIT     #ILR!RMR!UNS,RMER1I
055540  001514                            BEQ     16$

                                ;REPORT AN ERROR IF "ILR" IS SET
055542  032737  000002  001352            BIT     #ILR,RMER1I     ;WAS ILR SET DURING RECALIBRATE??
055550  001424                            BEQ     14$             ;NO!!
055552  013737  001352  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
055560  042737  000002  001140            BIC     #ILR,$GDDAT
055566  013737  001352  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
055574  062716  000004                    ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
055600  112776  000105  000000            MOVB    #105,@(SP)      ;WRITE ERROR NUMBER IN CALL
055606  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055612  004736                            JSR     PC,@(SP)+
055614  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
055620  000240                            NOP
055622                          14$:

                                ;REPORT AN ERROR IF "RMR" IS SET
055622  032737  000004  001352            BIT     #RMR,RMER1I     ;WAS RMR SET??
055630  001424                            BEQ     15$             ;NO!!
055632  013737  001352  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
055640  042737  000004  001140            BIC     #RMR,$GDDAT
055646  013737  001352  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
055654  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
055660  112776  000106  000000            MOVB    #106,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
055666  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055672  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
055674  162716  000010                    SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
055700  000240                            NOP
055702                          15$:

                                ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
055702  032737  040000  001352            BIT     #UNS,RMER1I     ;WAS UNSAFE ON??
055710  001430                            BEQ     16$             ;NO!!
055712  032737  000200  001400            BIT     #DVC,RMER2I     ;WAS THERE A DEVICE CHECK??
055720  001024                            BNE     16$             ;YES - DON'T REPORT UNSAFE
055722  013737  001352  001140            MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
055730  042737  040000  001140            BIC     #UNS,$GDDAT
055736  013737  001352  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
055744  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
055750  112776  000107  000000            MOVB    #107,@(SP)      ;WRITE ERROR NUMBER
055756  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
055762  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
055764  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
055770  000240                            NOP
055772                          16$:

                                ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
055772  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
055776  105776  000000                    TSTB    @(SP)           ;WAS AN ERROR REPORTED??
056002  001403                            BEQ     17$             ;NO!!
056004  062716  000004                    ADD     #4,(SP)         ;YES - AUGMENT SP RETURN
056010  000402                            BR      18$
056012  162716  000004            17$:    SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
```

```
    286 056016  000240                18$:    NOP
    287 056020  000207                        RTS     PC              ;STATUS CECK IS COMPLETE
```

```
                                .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
                                
056022                          DRVSTS:
                                
                                ;CLEAR USER'S ERROR CALL
056022  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056026  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
056032  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
                                ;REPORT ERROR IF RMCS1 NOT INITIALIZED
056036  013737  001336  001142 4$:       MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
056044  042737  173700  001142          BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
056052  012737  004010  001140          MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
056060  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
056066  001443                          BEQ     6$              ;BRANCH IF EQUAL
056070  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056074  112776  000141  000000          MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
056102  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
056106  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
056110  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
056114  000240                          NOP
                                ;REPORT ERROR IF RMDS NOT INITIALIZED
056116  013737  001350  001142 5$:       MOV     RMDSI,$BDDAT    ;CHECK RMDS
056124  042737  021101  001142          BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
056132  012737  010600  001140          MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
056140  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
056146  001413                          BEQ     6$              ;BRANCH IF EQUAL
056150  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056154  112776  000142  000000          MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
056162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
056166  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
056170  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
056174  000240                          NOP
                                ;REPORT ERROR IF RMER1 NOT INITIALIZED
056176  005037  001140 6$:       CLR     $GDDAT                  ;EXPECT 0'S
056202  013737  001352  001142          MOV     RMER1I,$BDDAT   ;CHECK RMER1
056210  001413                          BEQ     8$              ;BRANCH IF EQUAL
056212  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
056216  112776  000143  000000          MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
056224  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
056230  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
056232  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
056236  000240                          NOP
                                ;REPORT ERROR IF ATA NOT INITIALIZED
056240  013737  001354  001142 8$:       MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
056246  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
056250  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
056252  013701  001466                  MOV     TSTQUE,R1
056256  116102  000001                  MOVB    1(R1),R2
056262  042702  177400                  BIC     #^CATNMSK,R2
056266  005102                          COM     R2
056270  040237  001142                  BIC     R2,$BDDAT
```

```
     58 056274  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
     59 056276  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
     60 056300  005737  001142                  TST     $BDDAT          ;IS ATTENTION CLEARED??
     61 056304  001413                          BEQ     9$              ;BRANCH IF ATTENTION CLEARED
     62 056306  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
     63 056312  112776  000144  000000          MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
     64 056320  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     65 056324  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
     66 056326  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
     67 056332  000240                          NOP
     68                                 ;REPORT ERROR IF RMMR1 NOT INITIALIZED
     69 056334  013737  001362  001142  9$:     MOV     RMMR1I,$BDDAT   ;CHECK RMMR
     70 056342  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
     71 056350  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
     72 056356  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
     73 056364  001413                          BEQ     11$             ;BRANCH IF ZERO
     74 056366  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
     75 056372  112776  000145  000000          MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
     76 056400  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     77 056404  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
     78 056406  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
     79 056412  000240                          NOP
     80                                 ;REPORT ERROR IF RMMR2 NOT INITIALIZED
     81 056414  013737  001376  001142  11$:    MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
     82 056422  042737  140000  001142          BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
     83 056430  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
     84 056436  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
     85 056444  001413                          BEQ     15$             ;BRANCH IF EQUAL
     86 056446  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
     87 056452  112776  000146  000000          MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
     88 056460  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     89 056464  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
     90 056466  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
     91 056472  000240                          NOP
     92 056474  005037  001140          15$:    CLR     $GDDAT          ;EXPECT ZEROS
     93                                 ;REPORT ERROR IF RMEC2 NOT RESET
     94 056500  013737  001404  001142          MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
     95 056506  001413                          BEQ     17$             ;BRANCH IF 0
     96 056510  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
     97 056514  112776  000150  000000          MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
     98 056522  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     99 056526  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
    100 056530  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
    101 056534  000240                          NOP
    102                                 ;REPORT ERROR IF RMER2 NOT RESET
    103 056536  013737  001400  001142  17$:    MOV     RMER2I,$BDDAT   ;CHECK RMER2
    104 056544  001413                          BEQ     18$             ;BRANCH IF NO ERROR
    105 056546  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
    106 056552  112776  000147  000000          MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
    107 056560  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
    108 056564  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
    109 056566  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
    110 056572  000240                          NOP
    111 056574                         18$:
    112
    113 056574                         19$:
    114
```

```
                                        ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
056574  062716  000004                        ADD     #4,(SP)          ;MOVE SP TO ERROR CALL
056600  105776  000000                        TSTB    @(SP)            ;WAS AN ERROR DETECTED??
056604  001403                                BEQ     21$              ;NO!!
056606  062716  000004                        ADD     #4,(SP)          ;YES - MOVE SP TO ERROR RETURN
056612  000402                                BR      23$
056614  162716  000004              21$:      SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
056620  000240                      23$:      NOP
056622  000207                                RTS     PC               ;RETURN TO USER
```

```
                                .SBTTL  SEARCH STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEARCH OPERATIONS USING
                                ;STATUS STORED IN THE GET BUFFER AND TEST CONDITIONS STORED IN THE
                                ;PUT BUFFER.

                                ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS
                                ;DETECTED AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN
                                ;THE USER'S "ERROR" TRAP.

                                ;THE FOLLOWING CONDITIONS ARE CHECKED:
                                ;
                                ;       .ANY ERROR WHICH OCCURRED WHILE READING OR WRITING REMOTE
                                ;REGISTERS IS REPORTED, I.E., "MCPE" = 1 OR "PAR" = 1
                                ;
                                ;       ."IAE" STATUS IS CHECKED IN ACCORDANCE WITH THE VALUE DETERMINED
                                ;BY THE PROGRAM, WHICH IS BASED ON FORMAT AND ADDRESS.
                                ;
                                ;       ."OPI", IF SET, IS REPORTED AS 1) "OPI" DUE TO MOL = 0, OR 2)
                                ;"OPI" DUE TO ON CYLINDER LATCH.
                                ;
                                ;       ."IVC", IF SET, IS REPORTED AS 1) "IVC" ERROR WITH VOLUME
                                ;VALID ZERO, OR 2) ERRONEOUS "IVC" ERROR WITH VOLUME VALID SET.
                                ;
                                ;       ."SKI" IS REPORTED IF SET
                                ;
                                ;       ."DVC" IS REPORTED IF SET
                                ;
                                ;       .AN ERROR IS REPORTED IF "MOL" = 0, OR "PIP" = 1, OR "ATA" = 0,
                                ;OR "VV" = 0.
                                ;
                                ;CALL:
                                ;(1)    JSR     PC,SCHSTS
                                ;(2)    BR      ??                      RETURN HERE IF NO ERROR
                                ;(3)    NOP                             RETURN HERE TO REPORT ERROR
                                ;(4)    ERROR                           SUBROUTINE WILL LOAD ERROR #
                                ;(5)    JSR     PC,@(SP)+               GO BACK FOR MORE CHECKS
                                ;(6)    ??                              RETURN AFTER ALL ERRORS REPORTED

056624                          SCHSTS:

                                ;CLEAR USER'S ERROR CALL
056624  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
056630  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
056634  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR BR
056640  005037  060206                  CLR     200$            ;CLEAR STATUS FLAGS

                                ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING REMOTE
                                ;REGISTERS, I.E., "PAR" = 1 AND "DPE" = 0.
056644  032737  000010  001352          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
056652  001431                  BEQ     10$             ;NO!!
056654  032737  000010  001400          BIT     #DPE,RMER2I     ;WAS IT CONTROL BUS ERROR??
056662  001025                  BNE     10$             ;PROBABLY NOT!!

                                ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL
056664  013737  001352  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
056672  042737  000010  001140          BIC     #PAR,$GDDAT
```

SEARCH STATUS CHECK SUBROUTINE

```
 58 056700  013737  001352  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
 59 056706  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 60 056712  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
 61 056720  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
 62 056724  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
 63 056726  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 64 056732  000240                          NOP
 65 056734  000430                          BR      15$             ;SKIP FURTHER ERROR CHECKS
 66 056736                          10$:
 67
 68                                 ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN READING REMOTE
 69                                 ;REGISTERS, I.E., "MCPE" = 1.
 70 056736  032737  020000  001336          BIT     #MCPE,RMCS1I    ;WAS PARITY ERROR DETECTED??
 71 056744  001426                          BEQ     20$             ;NO!!
 72
 73                                 ;REPORT CONTROL BUS PARITY ERROR VIA USER'S ERROR CALL.
 74 056746  013737  001336  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
 75 056754  042737  020000  001140          BIC     #MCPE,$GDDAT
 76 056762  013737  001336  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
 77 056770  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 78 056774  112776  000013  000000          MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
 79 057002  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
 80 057006  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 81 057010  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 82 057014  000240                          NOP
 83 057016  000137  060160          15$:    JMP     150$            ;OMIT STATUS CHECKING
 84 057022                          20$:
 85
 86                                 ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK, SECTOR, CYLINDER
 87                                 ;AND THE STATUS OF "SSEI" DURING A SEARCH.  ALSO SET "SKI" FLAG IF
 88                                 ;CYLINDER ADDRESS IS TOO LARGE.
 89 057022  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR IAE = 1
 90 057030  052737  040000  060206          BIS     #SKI,200$       ;SETUP FOR SKI = 1
 91 057036  023727  001446  001060          CMP     RMDC0,#560.     ;GREATER THAN LAST CYLINDER ?
 92 057044  101035                          BHI     30$             ;YES - CYLINDER IS INVALID
 93 057046  042737  040000  060206          BIC     #SKI,200$       ;"SKI" SHOULD BE ZERO
 94
 95 057054  123737  001421  001335          CMPB    RMDA0+1,LSTRK+1 ;GREATER THAN LAST TRACK ?
 96 057062  101026                          BHI     30$             ;YES - TRACK IS INVALID
 97
 98 057064  123727  001420  000035          CMPB    RMDA0,#29.      ;SECTOR > 29. ?
 99 057072  101420                          BLOS    25$             ;NO!!
100 057074  032737  010000  001444          BIT     #FMT16,RMOF0    ;18 BIT FORMAT ?
101 057102  001416                          BEQ     30$             ;YES - SECTOR IS INVALID FOR 18 BIT MODE
102 057104  123727  001420  000036          CMPB    RMDA0,#30.      ;SECTOR > 30. ?
103 057112  101410                          BLOS    25$             ;BR IF NO
104 057114  032737  001000  001444          BIT     #SSEI,RMOF0     ;IS SSEI CLEAR ?
105 057122  001406                          BEQ     30$             ;YES - SECTOR IS INVALID FOR 16 BIT MODE
106 057124  123727  001420  000037          CMPB    RMDA0,#31.      ;SECTOR > 31. ?
107 057132  101002                          BHI     30$             ;YES - SECTOR IS INVALID
108
109 057134  005037  001140          25$:    CLR     $GDDAT          ;"IAE" SHOULD = 0
110
111                                 ;COMPARE EXPECTED AND RECIVED "IAE" STATUS
112 057140  013737  001352  001142  30$:    MOV     RMER1I,$BDDAT   ;GET RECEIVED IAE
113 057146  042737  175777  001142          BIC     #^CIAE,$BDDAT
114 057154  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS IAE CORRECT??
```

```
                                        BNE     40$                     ;NO!!
                                        BIC     #SKI,200$               ;SKI SHOULD BE ZERO
                                        BR      50$
                                40$:
                                ;REPORT INCORRECT IAE STATUS VIA USER'S ERROR CALL
                                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #257,@(SP)              ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)                ;RETURN SP TO NO ERROR
                                        NOP
                                50$:
                                ;SEE IF "SKI" OR "DVC" OR "IVC" IS SET
                                        BIT     #SKI!DVC!IVC,RMER2I     ;ARE ANY BITS SET??
                                        BEQ     90$                     ;NO!!
                                ;REPORT ERROR IF "SKI" IS SET AND "SKI" FLAG IS NOT SET
                                        MOV     RMER2I,$BDDAT           ;RECEIVED "SKI" STATUS
                                        BIC     #^CSKI,$BDDAT
                                        MOV     200$,$GDDAT             ;EXPECTED "SKI" STATUS
                                        BIC     #^CSKI,$GDDAT
                                        CMP     $GDDAT,$BDDAT           ;IS "SKI" OK??
                                        BEQ     60$                     ;YES-CHANGE ERROR NUMBER
                                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #263,@(SP)              ;WRITE ERROR NUMBER
                                        BIT     #SKI,$GDDAT             ;SHOULD "SKI" BE SET??
                                        BEQ     55$                     ;NO!!
                                        MOVB    #267,@(SP)              ;YES-CHANGE ERROR NUMBER
                                55$:    SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
                                        NOP
                                60$:
                                ;REPORT "IVC" ERROR AS
                                ;       .IVC DUE TO LOSS OF VOLUME VALID
                                ;       .ERRONEOUS IVC WITH VOLUME VALID SET
                                        BIT     #IVC,RMER2I             ;IS IVC SET??
                                        BEQ     80$                     ;NO!!
                                        MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
                                        BIC     #IVC,$GDDAT
                                        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
                                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #264,@(SP)              ;WRITE ERROR NUMBER IN CALL
                                        BIT     #VV,RMDSI               ;WAS VOLUME VALID??
                                        BEQ     70$                     ;NO!!
                                        MOVB    #265,@(SP)              ;YES - CHANGE ERROR NUMBER
                                70$:    SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
                                        NOP
                                80$:
                                ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
                                        BIT     #DVC,RMER2I             ;WAS THERE A DEVICE FAULT??
```

```
                        BEQ     90$                     ;NO!!
                        MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
                        BIC     #DVC,$GDDAT
                        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                        MOVB    #266,@(SP)              ;WRITE ERROR NUMBER IN CALL
                        SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                        SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
                        NOP
                90$:

                ;REPORT ANY "OPI" ERROR AS
                ;       ."OPI" BECAUSE MEDIUM IS NOT ONLINE
                ;       ."OPI" BECAUSE "ON CYLINDER" DIDN'T DROP
                        BIT     #OPI,RMER1I             ;WAS OPI SET??
                        BEQ     110$                    ;NO!!
                        MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
                        BIC     #OPI,$GDDAT
                        MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                        MOVB    #270,@(SP)              ;SETUP ERROR FOR MOL = 0
                        BIT     #MOL,RMDSI              ;WAS MOL 0??
                        BEQ     105$                    ;YES!!
                        MOVB    #271,@(SP)              ;MOL WAS 1 - CHANGE ERROR
                105$:   SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                        SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
                        NOP
                110$:

                ;SEE IF "ATA" = "MOL" = "VV" = 1 AND "PIP" = 0
                        MOV     RMDSI,-(SP)             ;GET DRIVE STATUS
                        BIC     #^C<ATA!PIP!MOL!VV>,(SP)
                        CMP     #ATA!MOL!VV,(SP)+       ;IS DRIVE STATUS CORRECT??
                        BEQ     150$                    ;YES!!

                ;REPORT AN ERROR IF MOL = 0 AND OPI ERROR WAS NOT REPORTED, I.E., OPI = 0
                        BIT     #MOL,RMDSI              ;WAS MEDIUM OFF LINE??
                        BNE     120$                    ;NO!!
                        MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
                        BIS     #MOL,$GDDAT
                        MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
                        BIT     #OPI,RMER1I             ;WAS OPI REPORTED BEFORE??
                        BNE     120$                    ;YES - DON'T REPORT MOL = 0
                        ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
                        MOVB    #272,@(SP)              ;WRITE ERROR NUMBER IN CALL
                        SUB     #2,(SP)                 ;MOVE SP TO RETURN IF ERROR
                        JSR     PC,@(SP)+               ;REPORT ERROR AND RETURN
                        SUB     #10,(SP)                ;RESTORE SP TO NO ERROR
                        NOP
                120$:

                ;REPORT AN ERROR IF PIP IS STIL SET AND SKI IS RESET
                        BIT     #PIP,RMDSI              ;IS POSITIONING IN PROGRESS??
                        BEQ     130$                    ;NO!!
                        BIT     #SKI,RMER2I             ;WAS "SKI" DETECTED??
```

```
                                        BNE     130$            ;YES-DONT REPORT PIP
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIC     #PIP,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #273,@(SP)      ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        NOP
                                130$:

                                ;REPORT AN ERROR IF VOLUME IS NOT VALID AND IVC = 0
                                        BIT     #VV,RMDSI       ;IS VOLUME VALID??
                                        BNE     140$            ;YES!!
                                        BIT     #IVC,RMER2I     ;WAS IVC DETECTED??
                                        BNE     140$            ;YES - DON'T REPORT VV = 0
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIS     #VV,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                                        MOVB    #350,@(SP)      ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        NOP
                                140$:

                                ;REPORT AN ERROR IF ATTENTION IS NOT SET
                                        BIT     #ATA,RMDSI      ;IS ATA ON??
                                        BNE     150$            ;YES!!
                                        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
                                        BIS     #ATA,$GDDAT
                                        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        MOVB    #351,@(SP)      ;WRITE ERROR NUMBER IN CALL
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
                                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
                                        SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                                        NOP
                                150$:

                                ;ARGUMENT THE RETURN ADDRESS IF AN ERROR WAS DETECTED
                                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                                        TSTB    @(SP)           ;WAS ERROR FOUND??
                                        BEQ     160$            ;NO!!
                                        ADD     #4,(SP)         ;YES - CHANGE RETURN
                                        BR      170$
                                160$:   SUB     #4,(SP)         ;NO ERROR FOUND
                                170$:   RTS     PC              ;RETURN TO USER

                                200$:   .WORD                   ;STORAGE FOR FLAGS
```

```
                        .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE

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

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

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

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

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

060210                  STCDRVSTS:

                        ;CLEAR USER'S ERROR CALL
060210  062716  000004          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
060214  105076  000000          CLRB    @(SP)                   ;CLEAR ERROR NUMBER
060220  162716  000004          SUB     #4,(SP)                 ;MOVE SP BACK TO NO ERROR RETURN
                        ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
060224  013746  001350          MOV     RMDSI,-(SP)             ;PUT DRIVE STATUS ON STACK
060230  042716  147677          BIC     #^C<PIP!MOL!VV>,(SP)
060234  022726  010100          CMP     #MOL!VV,(SP)+           ;ARE MOL,VV AND PIP O.K.??
060240  001524                  BEQ     30$                     ;YES!!

                        ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
060242  032737  010000  001350          BIT     #MOL,RMDSI              ;IS MOL ON ??
060250  001030                  BNE     10$                     ;YES!!
060252  032737  020000  001352          BIT     #OPI,RMER1I             ;WAS "OPI" SET??
060260  001024                  BNE     10$                     ;YES-DONT REPORT "MOL" = 0
060262  013737  001350  001140          MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
060270  052737  010000  001140          BIS     #MOL,$GDDAT
060276  013737  001350  001142          MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
060304  062716  000004          ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
060310  112776  000207  000000          MOVB    #207,@(SP)              ;WRITE ERROR NUMBER IN CALL
060316  162716  000002          SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
060322  004736                  JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
060324  162716  000010          SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR RETURN
060330  000240                  NOP
060332                  10$:

                        ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
060332  032737  000100  001350          BIT     #VV,RMDSI               ;IS "VV" = 0??
060340  001030                  BNE     20$                     ;NO!!
```

```
     58 060342  032737  010000  001400          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
     59 060350  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
     60 060352  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
     61 060360  052737  000100  001350          BIS     #VV,RMDSI
     62 060366  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
     63 060374  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
     64 060400  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
     65 060406  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     66 060412  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
     67 060414  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
     68 060420  000240                          NOP
     69 060422                          20$:
     70
     71                                 ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
     72 060422  032737  020000  001350          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
     73 060430  001430                          BEQ     30$             ;NO!!
     74 060432  032737  040000  001400          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
     75 060440  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
     76 060442  013737  001350  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
     77 060450  042737  020000  001140          BIC     #PIP,$GDDAT
     78 060456  013737  001350  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
     79 060464  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
     80 060470  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
     81 060476  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
     82 060502  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
     83 060504  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
     84 060510  000240                          NOP
     85 060512                          30$:
     86
     87                                 ;SEE IF "SKI" = "DVC" = 0
     88 060512  013746  001400                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
     89 060516  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
     90 060522  001460                          BEQ     60$             ;BRANCH IF NO ERROR
     91 060524                          40$:
     92
     93                                 ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
     94 060524  032737  000200  001400          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
     95 060532  001424                          BEQ     50$             ;NO!!
     96 060534  013737  001400  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
     97 060542  042737  000200  001140          BIC     #DVC,$GDDAT
     98 060550  013737  001400  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
     99 060556  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
    100 060562  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
    101 060570  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
    102 060574  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
    103 060576  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
    104 060602  000240                          NOP
    105 060604                          50$:
    106
    107                                 ;REPORT AN ERROR IF "SKI" = 1
    108 060604  032737  040000  001400          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
    109 060612  001424                          BEQ     60$             ;NO!!
    110 060614  013737  001400  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
    111 060622  042737  040000  001140          BIC     #SKI,$GDDAT
    112 060630  013737  001400  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
    113 060636  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
    114 060642  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
```

```
    115 060650  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
    116 060654  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
    117 060656  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
    118 060662  000240                          NOP
    119 060664                          60$:
    120
    121                                 ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
    122 060664  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
    123 060670  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
    124 060674  001403                          BEQ     70$             ;NO!!
    125 060676  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
    126 060702  000402                          BR      80$
    127 060704  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
    128 060710  000240                  80$:    NOP
    129 060712  000207                          RTS     PC              ;RETURN TO USER
```

```
     1                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                        ;;*****************************************************************
                                        ;*SAVE R0-R5
                                        ;*CALL:
                                        ;*      SAVREG
                                        ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                        ;*
                                        ;*TOP---(+16)
                                        ;* +2---(+18)
                                        ;* +4---R5
                                        ;* +6---R4
                                        ;* +8---R3
                                        ;*+10---R2
                                        ;*+12---R1
                                        ;*+14---R0

060714                                  $SAVREG:
060714  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
060716  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
060720  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
060722  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
060724  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
060726  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
060730  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
060734  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
060740  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
060744  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
060750  000002                                  RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
060752                                  $RESREG:
060752  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
060756  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
060762  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
060766  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
060772  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
060774  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
060776  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
061000  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
061002  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
061004  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
061006  000002                                  RTI
```

```
                                .SBTTL  BINARY TO ASCII AND TYPE ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
                                ;*BINARY-ASCII NUMBER AND TYPE IT.
                                ;*CALL:
                                ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
                                ;*      TYPBN                   ;;TYPE IT

061010  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
061012  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
061016  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
061020  112737  000060  061062  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
061026  006101                          ROL     R1              ;;GET THIS BIT
061030  001406                          BEQ     2$              ;;DONE?
061032  105537  061062                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
061036  104401  061062                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
061042  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
061044  000765                          BR      1$              ;;GO DO THE NEXT BIT
061046  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
061050  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
061056  012616                          MOV     (SP)+,(SP)
061060  000002                          RTI                     ;;RETURN TO USER
061062     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
```

```
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE

061064                          $TYPDS:
061064  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
061066  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
061070  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
061072  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
061074  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
061076  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
061102  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
061106  100004                          BPL     1$              ;;BR IF INPUT IS POS.
061110  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
061112  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
061120  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
061122  012703  061300                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
061126  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
061132  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
061134  016001  061270                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
061140  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
061142  002402                          BLT     4$              ;;BR IF DONE
061144  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
061146  000774                          BR      3$
061150  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
061152  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
061154  001002                          BNE     5$              ;;FALL THROUGH IF 0
061156  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
061160  100407                          BMI     7$              ;;BR IF YES
061162  106316                  5$:     ASLB    (SP)            ;;MSD?
061164  103003                          BCC     6$              ;;BR IF NO
061166  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
061174  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
061200  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
061204  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
061206  005720                          TST     (R0)+           ;;JUST INCREMENTING
061210  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
061214  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
061216  003002                          BGT     8$              ;;GO TO EXIT
061220  010502                          MOV     R5,R2           ;;GET THE LSD
061222  000764                          BR      6$              ;;GO CHANGE TO ASCII
061224  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
061226  100003                          BPL     9$              ;;BR IF NO
061230  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
061236  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
061240  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
061242  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
061244  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
061246  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
```

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```
061250  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
061252  104401  061300                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
061256  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
061264  012616                          MOV     (SP)+,(SP)
061266  000002                          RTI                     ;;RETURN TO USER
061270  023420                  $DTBL:  10000.
061272  001750                          1000.
061274  000144                          100.
061276  000012                          10.
061300                          $DBLK:  .BLKW   4
```

```
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                ;;*****************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT

061310  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
061314  116637  000001  061533          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
061322  112637  061535                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
061326  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
061332  000406                          BR      $TYPON
061334  112737  000001  061533  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
061342  112737  000006  061535          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
061350  112737  000005  061532  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
061356  010346                          MOV     R3,-(SP)        ;;SAVE R3
061360  010446                          MOV     R4,-(SP)        ;;SAVE R4
061362  010546                          MOV     R5,-(SP)        ;;SAVE R5
061364  113704  061535                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
061370  005404                          NEG     R4
061372  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
061376  110437  061534                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
061402  113704  061533                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
061406  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
061412  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
061414  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
061416  000404                          BR      3$              ;;GO DO MSB
061420  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
061422  006105                          ROL     R5
061424  006105                          ROL     R5
061426  010503                          MOV     R5,R3
061430  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
061432  105337  061534                  DECB    $OMODE          ;;TYPE THIS DIGIT?
061436  100016                          BPL     7$              ;;BR IF NO
061440  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
061444  001002                          BNE     4$              ;;TEST FOR 0
061446  005704                          TST     R4              ;;SUPPRESS THIS 0?
061450  001403                          BEQ     5$              ;;BR IF YES
061452  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
061454  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
061460  052703  000040     5$:    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
061464  110337  061530            MOVB    R3,8$           ;;SAVE FOR TYPING
061470  104401  061530            TYPE    ,8$             ;;GO TYPE THIS DIGIT
061474  105337  061532     7$:    DECB    $OCNT           ;;COUNT BY 1
061500  003347                    BGT     2$              ;;BR IF MORE TO DO
061502  002402                    BLT     6$              ;;BR IF DONE
061504  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
061506  000744                    BR      2$              ;;GO DO THE LAST DIGIT
061510  012605             6$:    MOV     (SP)+,R5        ;;RESTORE R5
061512  012604                    MOV     (SP)+,R4        ;;RESTORE R4
061514  012603                    MOV     (SP)+,R3        ;;RESTORE R3
061516  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
061524  012616                    MOV     (SP)+,(SP)
061526  000002                    RTI                     ;;RETURN
061530     000             8$:    .BYTE   0               ;;STORAGE FOR ASCII DIGIT
061531     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
061532     000             $OCNT: .BYTE   0               ;;OCTAL DIGIT COUNTER
061533     000             $OFILL:.BYTE   0               ;;ZERO FILL SWITCH
061534  000000             $OMODE:.WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TYPE ROUTINE

                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

061536  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
061542  100002                          BPL     1$              ;;BR IF YES
061544  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
061546  000430                          BR      3$              ;;LEAVE
061550  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
061552  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
061556  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
061564  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
061566  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
061574  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
061576  010037  061606                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
061602  004737  066260                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
061606  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
061610  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
061616  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
061620  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
061622  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
061624  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
061626  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
061630  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
061634  000002                          RTI                     ;;RETURN
061636  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
061642  001430                          BEQ     8$
061644  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
061650  001006                          BNE     5$
061652  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
061654  104401                          TYPE                    ;;TYPE A CR AND LF
061656  001217                          $CRLF
061660  105037  062066                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
061664  000755                          BR      2$              ;;GET NEXT CHARACTER
061666  004737  061750          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
061672  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
061676  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
061700  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
061704  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
061710  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
061712  004737  061750                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
061716  105337  062066                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
061722  000770                          BR      7$              ;;LOOP
```

```
                                        ;HORIZONTAL TAB PROCESSOR

061724  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
061730  004737  061750          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
061734  132737  000007  062066          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
061742  001372                          BNE     9$              ;;TAB STOP
061744  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
061746  000724                          BR      2$              ;;GET NEXT CHARACTER
061750                          $TYPEC:
061750  105777  117204                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
061754  100022                          BPL     10$             ;;BR IF NOT
061756  017746  117200                  MOV     @$TKB,-(SP)     ;;GET CHAR
061762  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
061766  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
061772  001012                          BNE     102$            ;;BR IF NOT
061774                          101$:
061774  105777  117160                  TSTB    @$TKS           ;;WAIT FOR CHAR
062000  100375                          BPL     101$
062002  117716  117154                  MOVB    @$TKB,(SP)      ;;GET CHAR
062006  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT
062012  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?
062016  001366                          BNE     101$            ;;BR IF NOT
062020                          102$:
062020  005726                  10$:    TST     (SP)+           ;;FIX STACK
062022
062022  105777  117136                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
062026  100375                          BPL     10$
062030  116677  000002  117130          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
062036  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
062044  001003                          BNE     1$              ;;BRANCH IF NO
062046  105037  062066                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
062052  000406                          BR      $TYPEX          ;;EXIT
062054  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
062062  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
062064  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
062066  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
062070  000207                  $TYPEX: RTS     PC
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

062072                          $SCOPE:
062072  104410                         CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
062074  004737  062734                  JSR     PC,STOP
062100  032777  040000  117046  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
062106  001402                          BEQ     9$              ;;NO IF SW14=0
062110  000137  062540                  JMP     $OVER           ;;JUMP OVER SCOPE ROUTINE
062114                          9$:
                                ;#####START OF CODE FOR THE XOR TESTER#####
062114  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
062116  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
062122  012737  062142  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
062130  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
062134  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
062140  000561                          BR      $SVLAD          ;;GO TO THE NEXT TEST
062142  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
062144  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
062150  000521                          BR      7$              ;;LOOP ON THE PRESENT TEST
062152                          6$:;#####END OF CODE FOR THE XOR TESTER#####
062152  032777  000400  116774          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
062160  001421                          BEQ     2$              ;;BR IF NO
062162  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
062164  117716  116764                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
062170  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
062172  022716  000067                  CMP     #67,(SP)        ;;CHECK THE NUMBER IN THE SWR
062176  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
062200  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
062204  005316                          DEC     (SP)            ;;BACKUP BY ONE
062206  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
062210  062716  062556                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
062214  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
062220  000547                          BR      $OVER           ;;GO LOOP ON THE TEST
062222  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
062224  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
062230  001502                          BEQ     3$              ;;BR IF NO
062232  022737  177777  063352          CMP     #-1,CPSAVE      ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
062240  001455                          BEQ     2003$           ;;KICK AROUND ROUTINE IF SO
062242  013746  000004                  MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
062246  012737  062264  000004          MOV     #2000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
062254  013737  177766  063352          MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
062262  000406                          BR      2001$
062264  012737  177777  063352  2000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
062272  012716  062300                  MOV     #2001$,(SP)     ;;SETUP RETURN ADDRESS
```

```
062276  000002                          RTI
062300  012637  000004          2001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

062304  022737  177777  063352  2002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
062312  001430                          BEQ     2003$           ;;BRANCH IF SO
062314  032737  000001  063352          BIT     #BIT00,CPSAVE   ;;SEE IF THE POWER MONITOR BIT IS ON
062322  001424                          BEQ     2003$           ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
062324  042737  000001  177766          BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND TO BE SET
062332  013746  001154                  MOV     SWR,-(SP)       ;;SAVE SWR ADDRESS
062336  017646  000000                  MOV     @(SP),-(SP)     ;;SAVE SWR VALUE
062342  012737  000176  001154          MOV     #176,SWR        ;;GET SOFTWARE SWR ADDRESS
062350  011677  116600                  MOV     (SP),@SWR       ;;GET CURRENT SWR VALUE
062354  042777  001000  116572          BIC     #BIT09,@SWR     ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
062362  104177                          EMT     177             ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
062364  012676  000000                  MOV     (SP)+,@(SP)     ;;RESTORE SWR TO ORIGINAL VALUE
062370  012637  001154                  MOV     (SP)+,SWR       ;;RESTORE SWR ADDRESS
062374                          2003$:
062374  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
062402  101015                          BHI     3$              ;;BR IF NO
062404  032777  001000  116542          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
062412  001404                          BEQ     4$              ;;BR IF NO
062414  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
062422  000446                          BR      $OVER
062424  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
062430  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
062434  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
062436  032777  004000  116510  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
062444  001011                          BNE     1$              ;;BR IF YES
062446  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
062452  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
062454  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
062460  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
062466  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
062470  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
062476  013737  062554  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
062504  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
062510  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
062516  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
062522  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
062526  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
062532  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
062540  013777  001116  116410  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
062546  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
062552  000002                          RTI                     ;;FIXES PS
062554  000005                  $MXCNT: 5.                      ;;MAX. NUMBER OF ITERATIONS
062556                          $SW08TBL:
        000067                  .REPT   $TN-1
062556  007740                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
062560  010140                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
062562  010534                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
062564  011404                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
062566  011742                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
062570  012150                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
062572  012464                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
062574  012644                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
062576  013716                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
062600  014216                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
```

```
062602  014426                  .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
062604  014734                  .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
062606  015244                  .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
062610  015506                  .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
062612  015764                  .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
062614  016234                  .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
062616  016464                  .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
062620  017010                  .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
062622  017430                  .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
062624  017774                  .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
062626  020366                  .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
062630  021056                  .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
062632  021424                  .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
062634  021746                  .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
062636  022330                  .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
062640  022646                  .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
062642  023432                  .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
062644  024156                  .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
062646  024746                  .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
062650  025532                  .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
062652  026064                  .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
062654  026524                  .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
062656  026744                  .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
062660  027302                  .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
062662  027574                  .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
062664  030066                  .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
062666  030400                  .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
062670  030706                  .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
062672  031174                  .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
062674  031476                  .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
062676  032002                  .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
062700  032420                  .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
062702  033016                  .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
062704  033410                  .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
062706  033750                  .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
062710  034312                  .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
062712  034636                  .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
062714  035356                  .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
062716  036000                  .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
062720  036442                  .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
062722  037042                  .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
062724  037422                  .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
062726  037776                  .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
062730  040356                  .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
062732  040722                  .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
2
3                       ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
4
5 062734                STOP:
  062734  012746  000140        MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
  062740  012746  062746        MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
  062744  000002                RTI                     ;;POP NEW PC AND PS
  062746                64$:
6
7                       ;RAISE PRIORITY TO INHIBIT CONSOLE INTERRUPT
8
9 062746  012746  000240        MOV     #PR5,-(SP)      ;;PUT NEW PS ON STACK
```

```
062752  012746  062760                  MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
062756  000002                          RTI                     ;;POP NEW PC AND PS
062760                          65$:
062760  000207                          RTS     PC              ;RETURN
```

```
     1                                  .SBTTL  ERROR HANDLER ROUTINE

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

062762  105037  063354          $ERROR: CLRB    IBSAVE          ;;CLEAR THE ITEM BYTE SAVE LOCATION
062766  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
062770  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
062774  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
062776  013777  001116  116152          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
063004  032777  002000  116142          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
063012  001402                          BEQ     1$              ;;NO - SKIP
063014  104401  001212                  TYPE    ,$BELL          ;;RING BELL
063020  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
063024  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
063030  162737  000002  001132          SUB     #2,$ERRPC
063036  117737  116070  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
063044  032777  001000  116102          BIT     #BIT09,@SWR     ;;SEE IF LOOP ON ERROR IS SET
063052  001060                          BNE     1004$           ;;BRANCH AROUND ROUTINE IF SO
063054  122737  000177  001130          CMPB    #177,$ITEMB     ;;SEE IF THIS IS THE POWER FAIL CALL
063062  001454                          BEQ     1004$           ;;BRANCH AROUND ROUTINE IF IT IS
063064  105737  063354                  TSTB    IBSAVE          ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
063070  001047                          BNE     1003$           ;;BRANCH IF SO
063072  022737  177777  063352          CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
063100  001445                          BEQ     1004$           ;;BRANCH IF SO
063102  013746  000004                  MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
063106  012737  063124  000004          MOV     #1000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
063114  013737  177766  063352          MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
063122  000406                          BR      1001$
063124  012737  177777  063352  1000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
063132  012716  063140                  MOV     #1001$,(SP)     ;;SETUP RETURN ADDRESS
063136  000002                          RTI
063140  012637  000004          1001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

063144  022737  177777  063352  1002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
063152  001420                          BEQ     1004$           ;;BRANCH IF SO
063154  032737  000001  063352          BIT     #BIT00,CPSAVE   ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
063162  001414                          BEQ     1004$           ;;BRANCH IF OK
063164  042737  000001  177766          BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND SET
063172  113737  001130  063354          MOVB    $ITEMB,IBSAVE   ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
063200  112737  000177  001130          MOVB    #177,$ITEMB     ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
063206  000402                          BR      1004$           ;;BRANCH OVER IBSAVE CLEARING

063210  105037  063354          1003$:  CLRB    IBSAVE          ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
063214                          1004$:
063214  032777  020000  115732          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
063222  001004                          BNE     20$             ;;SKIP TYPEOUTS
063224  004737  063356                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
```

```
063230  104401  001217          TYPE    ,$CRLF
063234                  20$:
063234  122737  000001  001242  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
063242  001007                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
063244  113737  001130  063256  MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
063252  004737  066270          JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
063256     000          21$:    .BYTE   0
063257     000          22$:    .BYTE   0
063260  000777          2$:     BR      22$             ;;APT ERROR LOOP
063262  105737  063354          TSTB    IBSAVE          ;;SEE IF IBSAVE IS LOADED
063266  001005                  BNE     3$              ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
063270  005777  115660          TST     @SWR            ;;HALT ON ERROR
063274  100002                  BPL     3$              ;;SKIP IF CONTINUE
063276  000000                  HALT                    ;;HALT ON ERROR!
063300  104410                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
063302                  3$:
063302  032777  001000  115644  BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
063310  001402                  BEQ     4$              ;;BR IF NO
063312  013716  001124          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
063316  005737  001210  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
063322  001402                  BEQ     5$              ;;BR IF NONE
063324  013716  001210          MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
063330                  5$:
063330  022737  041512  000042  CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
063336  001001                  BNE     6$              ;;BRANCH IF NO
063340  000000                  HALT                    ;;YES
063342                  6$:
063342  105737  063354          TSTB    IBSAVE          ;;SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
063346  001210                  BNE     7$              ;;BRANCH BACK TO CALL ORIGINAL ERROR
063350  000002                  RTI                     ;;RETURN
063352  000000          CPSAVE: .WORD   0               ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
063354  000000          IBSAVE: .WORD   0               ;;LOCATION TO SAVE ITEM BYTE
```

ERROR TYPEOUT ROUTINE

```
                                        .SBTTL  ERROR TYPEOUT ROUTINE
                                ;
                                ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
                                ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
                                ;*
                                ;*      .UNIT NUMBER, DRIVE TYPE, 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.
                                ;
063356  104414                  ERRTYP: SAVREG
063360  032777  020000  115566          BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
063366  001402                        BEQ     1$              ;NO!!
063370  000137  064166                JMP     27$             ;YES!!
                                ;
                                ;TYPE UNIT NUMBER, DRIVE TYPE, TEST NUMBER, ERROR NUMBER, AND
                                ;PROGRAM COUNTER
063374  104401  001217          1$:     TYPE    ,$CRLF
063400  104401  064202                  TYPE    ,ERTY00         ;TYPE "DRV#"
063404  013746  001234                  MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
063410  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
063412     003                        .BYTE   3               ;;TYPE 3 DIGIT(S)
063413     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                ;
                                ;TYPE 'DRIVE TYPE', RM80 FOR UNIT UNDER TEST
063414  013700  001276                  MOV     $BASE,R0        ;GET RM BASE ADDRESS
063420  016000  000026                  MOV     RMDT(R0),R0     ;GET DRIVE TYPE REGISTER
063424  042700  177740                  BIC     #177740,R0      ;SAVE DRIVE TYPE BITS AND
063430  012737  067130  063452          MOV     #$RM80,3$       ;GET ASCII DRIVE TYPE
063436  022700  000026                  CMP     #26,R0          ;IS DEVICE AN RM80 ?
063442  001004                        BNE     4$              ;NO !!
063444  104401  064237          2$:     TYPE    ,ERTY05         ;TYPE " - "
063450  104401                        TYPE                    ;TYPE DRIVE TYPE
063452  000000                  3$:     .WORD   0               ;DRIVE TYPE MESSAGE IS STORED HERE
                                ;
                                ;TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
063454  005037  064172          4$:     CLR     TSTNMB          ;LOAD TEST NUMBER FOR
063460  013737  001226  064172        MOV     $TESTN,TSTNMB
063466  104401  064207                TYPE    ,ERTY01         ;TYPE "TST#"
063472  013746  064172                MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                ;;TYPE TEST NUMBER
063476  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
063500     003                        .BYTE   3               ;;TYPE 3 DIGIT(S)
063501     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
063502  005037  064174                CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
063506  113737  001130  064174        MOVB    $ITEMB,ERRNMB   ;TYPEOUT
063514  001406                        BEQ     5$              ;SKIP IF NO ERROR CALLED
063516  104401  064217                TYPE    ,ERTY02         ;TYPE "ERR#"
063522  013746  064174                MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                ;;TYPE ERROR NUMBER
063526  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
063530     003                        .BYTE   3               ;;TYPE 3 DIGIT(S)
063531     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
063532  104401  064226          5$:     TYPE    ,ERTY03         ;TYPE "PC="
```

ERROR TYPEOUT ROUTINE

```
 46 063536  013746  001132                MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                    ;;TYPE PROGRAM COUNTER
    063542  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
    063544     006                        .BYTE   6               ;;TYPE 6 DIGIT(S)
    063545     001                        .BYTE   1               ;;TYPE LEADING ZEROS
 47
 48                                     ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
 49 063546  005737  064174             6$:     TST     ERRNMB          ;WAS AN ERROR CALLED?
 50 063552  001002                             BNE     7$              ;BR IF YES
 51 063554  000137  064166                     JMP     27$             ;NO--EXIT
 52
 53 063560  104401  001217             7$:     TYPE    ,$CRLF          ;YES-TYPE CRLF
 54 063564  105037  064200                     CLRB    BOTFLG          ;CLEAR BOT FLAG
 55 063570  105037  064201                     CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
 56 063574  013700  064174                     MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
 57 063600  122700  000177                     CMPB    #177,R0         ;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
 58 063604  001003                             BNE     8$              ;BRANCH IF NOT
 59 063606  012700  064244                     MOV     #PFECH,R0       ;MOVE POWER FAIL ERROR CALL TABLE TO R0
 60 063612  000405                             BR      9$
 61 063614  006300                     8$:     ASL     R0              ;FOUR ENTRIES IN ERROR
 62 063616  006300                             ASL     R0              ;TABLE
 63 063620  006300                             ASL     R0
 64 063622  062700  001564                     ADD     #$ERRTB-8.,R0
 65 063626  011001                     9$:     MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
 66                                                                 ;TABLE
 67 063630  001507                             BEQ     19$             ;BRANCH IF NO ERROR MESSAGE
 68
 69                                     ;TYPE THE ERROR MESSAGE
 70 063632  012102                     10$:    MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
 71 063634  001505                             BEQ     19$             ;BRANCH IF END OF MESSAGE
 72 063636  010237  064004                     MOV     R2,18$          ;LOAD ADDRESS OF STRING
 73 063642  005037  064176                     CLR     BOTADR          ;CLEAR BOT ADDRESS
 74 063646  112203                     11$:    MOVB    (R2)+,R3        ;END OF STRING??
 75 063650  001454                             BEQ     17$             ;YES!!
 76 063652  122703  000015                     CMPB    #CR,R3          ;CARRIAGE RETURN??
 77 063656  001003                             BNE     12$             ;NO!!
 78 063660  105037  064201                     CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
 79 063664  000770                             BR      11$             ;GET NEXT CHARACTER
 80 063666  122703  000012                12$: CMPB    #LF,R3          ;LINE FEED??
 81 063672  001765                             BEQ     11$             ;YES-GET NEXT CHARACTER
 82 063674  122703  000011                     CMPB    #HT,R3          ;HORIZONTAL TAB??
 83 063700  001007                             BNE     14$             ;NO!!
 84 063702  105237  064201                13$: INCB    CHRCNT          ;ADJUST CHARACTER COUNT
 85 063706  132737  000007  064201             BITB    #7,CHRCNT
 86 063714  001372                             BNE     13$
 87 063716  000407                             BR      15$
 88 063720  105237  064201                14$: INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
 89 063724  122703  000040                     CMPB    #' ,R3          ;SPACE??
 90 063730  001002                             BNE     15$             ;NO!!
 91 063732  010237  064176                     MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
 92 063736  122737  000100  064201        15$: CMPB    #64.,CHRCNT     ;END OF LINE??
 93 063744  103340                             BHIS    11$             ;NO!!
 94 063746  013704  064176                     MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
 95 063752  001007                             BNE     16$             ;BRANCH IF SPACE DETECTED
 96 063754  104401  001217                     TYPE    ,$CRLF          ;TYPE CRLF
 97 063760  105037  064201                     CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 98 063764  013702  064004                     MOV     18$,R2          ;SET UP R2 FOR TESTING
```

ERROR TYPEOUT ROUTINE

```
 063770  000726                          BR      11$
 063772  105044                  16$:    CLRB    -(R4)           ;REPLACE SPACE
 063774  112737  177777  064200          MOVB    #-1,BOTFLG      ;SET BOT FLAG
 064002  104401                  17$:    TYPE                    ;TYPE ERROR MESSAGE STRING
 064004  000000                  18$:    .WORD                   ;STRING ADDRESS GOES HERE
 064006  105737  064200                  TSTB    BOTFLG          ;WAS STRING TRUNCATED??
 064012  001707                          BEQ     10$             ;NO!!
 064014  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
 064020  105037  064200                  CLRB    BOTFLG          ;CLEAR BOT FLAG
 064024  105037  064201                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 064030  013702  064176                  MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
 064034  010237  064004                  MOV     R2,18$          ;SETUP 18$ FOR TYPING
 064040  112742  000040                  MOVB    #' ,-(R2)       ;RESTORE SPACE
 064044  105722                          TSTB    (R2)+           ;RESTORE R2
 064046  000677                          BR      11$             ;TYPE REST OF STRING

                         ;TYPE ERROR HEADER AND ERROR DATA
 064050  016001  000002          19$:    MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
 064054  001444                          BEQ     27$             ;BRANCH IF NO HEADER
 064056  104401  001217                  TYPE    ,$CRLF          ;(ASSUME NO DATA)
 064062  016002  000004                  MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
 064066  016003  000006                  MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
 064072  012137  064102          20$:    MOV     (R1)+,21$       ;PUT HEADER ADDRESS FOR TYPE
 064076  001433                          BEQ     27$             ;BRANCH IF END OF HEADERS
                                                                 ;(ASSUME END OF DATA)
 064100  104401                          TYPE
 064102  000000                  21$:    .WORD   0               ;HEADER ADDRESS GOES HERE
 064104  104401  001217                  TYPE    ,$CRLF
 064110  005702                          TST     R2              ;DATA WITH HEADER??
 064112  001767                          BEQ     20$             ;NO!!
 064114  012204                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
 064116  012305                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
 064120  105725                  22$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
 064122  100407                          BMI     24$             ;BINARY
 064124  001403                          BEQ     23$             ;OCTAL
 064126  013446                          MOV     @(R4)+,-(SP)    ;DECIMAL
 064130  104405                          TYPDS
 064132  000405                          BR      25$
 064134  013446                  23$:    MOV     @(R4)+,-(SP)
 064136  104402                          TYPOC
 064140  000402                          BR      25$
 064142  013446                  24$:    MOV     @(R4)+,-(SP)
 064144  104406                          TYPBN
 064146  005714                  25$:    TST     (R4)            ;MORE DATA??
 064150  001403                          BEQ     26$             ;NO!!
 064152  104401  064234                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
 064156  000760                          BR      22$             ;AND CONTINUE
 064160  104401  001217          26$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
 064164  000742                          BR      20$             ;BEFORE NEXT HEADER
 064166  104415                  27$:    RESREG
 064170  000207                          RTS     PC

 064172  000000          TSTNMB: .WORD   0               ;TEST NUMBER
 064174  000000          ERRNMB: .WORD   0               ;ERROR NUMBER
 064176  000000          BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
 064200      000         BOTFLG: .BYTE   0               ;BOT FLAG
 064201      000         CHRCNT: .BYTE   0               ;CHARACTER COUNT
```

```

    064202     104     122     126  ERTY00: .ASCIZ  @DRV#@
    064207     054     040     124  ERTY01: .ASCIZ  @, TEST#@
    064217     054     040     105  ERTY02: .ASCIZ  @, ERR#@
    064226     054     040     120  ERTY03: .ASCIZ  @, PC=@
    064234     040     040     000  ERTY04: .ASCIZ  @ @
    064237     040     055     040  ERTY05: .ASCIZ  @ - @
                                    .EVEN
    064244  064254  064342  064360  PFECH:  PFECH1,PFECH2,PFECH3,PFECH4 ;WORDS DEFINING TABLES BELOW
    064254  064260  000000          PFECH1: .+4,0
    064260     120     117     127          .ASCIZ  ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?
                                            .EVEN
    064342  064346  000000          PFECH2: .+4,0
    064346     103     120     125          .ASCIZ  ?CPUERREG?
                                            .EVEN
    064360  064362                  PFECH3: .+2
    064362  063352  000000                  .WORD   CPSAVE,0
    064366  064370                  PFECH4: .+2
    064370     000     000                  .BYTE   0,0
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*****************************************************************************
                                .ENABL  LSB
064372  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
064374  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
064376  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
064400                          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        064401                  $TKQEND=.
                                .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
064402  005037  064372          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
064406  012737  064400  064374          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
064414  013737  064374  064376          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
064422  012737  064452  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
064430  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
064436  005777  114520                  TST     @$TKB           ;;CLEAR DONE FLAG
064442  012777  000100  114510          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
064450  000207                          RTS     PC              ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT)
                                ;
064452  117746  114504          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
064456  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
064462  021627  000021                  CMP     (SP),#$XON      ;;IS IT A RANDOM XON?
064466  001002                          BNE     30$             ;;BRANCH IF NO
064470  005726                          TST     (SP)+           ;;CLEAN RANDOM XON OFF STACK
064472  000002                          RTI                     ;;RETURN
064474                          30$:
064474  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
064500  001007                          BNE     1$              ;;BRANCH IF NO
064502  104401  065600                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
064506  004737  064402                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
064512  005726                          TST     (SP)+           ;;CLEAN UP STACK
064514  000137  065642                  JMP     SHUT            ;;CONTROL C RESTART
064520  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
064524  001004                          BNE     2$              ;;BRANCH IF NO
064526  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
064534  001500                          BEQ     6$              ;;GO TO SWR CHANGE

064536                          2$:
064536  022737  000001  064372          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
064544  001004                          BNE     3$              ;;BRANCH IF NO
064546  104401  001212                  TYPE    ,$BELL          ;;RING THE TTY BELL
```

```
064552  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
064554  000451                          BR      5$              ;;EXIT
064556  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
064562  001021                          BNE     32$             ;;BRANCH IF NO
064564  005077  114370                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
064570  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
064572  105777  114362          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
064576  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
064600  117746  114356                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
064604  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
064610  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
064614  001366                          BNE     31$             ;;BRANCH IF NO
064616  012777  000100  114334          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
064624  000002                          RTI                     ;;RETURN
064626  005237  064372          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
064632  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
064636  002405                          BLT     4$              ;;BRANCH IF YES
064640  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
064644  003002                          BGT     4$              ;;BRANCH IF YES
064646  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
064652  112677  177516          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
064656  005237  064374                  INC     $TKQIN          ;;UPDATE THE POINTER
064662  023727  064374  064401          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
064670  001003                          BNE     5$              ;;BRANCH IF NO
064672  012737  064400  064374          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
064700  000002                  5$:     RTI                     ;;RETURN

                                ;;**************************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
064702  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
064710  001124                          BNE     15$             ;;EXIT IF NOT
064712  105777  114242                  TSTB    @$TKS           ;;IS A CHAR WAITING?
064716  100121                          BPL     15$             ;;IF NOT, EXIT
064720  117746  114236                  MOVB    @$TKB,-(SP)     ;;YES
064724  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
064730  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
064734  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                ;;**************************************************************************
                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
064736  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
064744  001674                          BEQ     2$              ;;BRANCH IF YES
064746  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
064750  004737  064402                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
064754  005077  114200                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
064760  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

064766  104401  065612                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
064772  104401  065617          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
064776  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
065002  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
065004  104401  065630                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
065010  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
065012  005046                          CLR     -(SP)           ;;THE NEW SWR
065014  105777  114140          7$:     TSTB    @$TKS           ;;CHAR THERE?
065020  100375                          BPL     7$              ;;IF NOT TRY AGAIN

065022  117746  114134                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
065026  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

065032  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
065036  001015                          BNE     9$              ;;BRANCH IF NOT
065040  104401  065600                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
065044  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
065050  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
065056  001003                          BNE     8$              ;;BRANCH IF NO
065060  012777  000100  114072          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
065066  000137  065642          8$:     JMP     SHUT            ;;CONTROL-C RESTART


065072  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
065076  001005                          BNE     10$             ;;BRANCH IF NOT
065100  104401  065605                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
065104  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
065110  000737                          BR      19$             ;;LET'S TRY IT AGAIN


065112  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
065116  001022                          BNE     16$             ;;BRANCH IF NO
065120  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
065124  001403                          BEQ     11$             ;;BRANCH IF YES
065126  016677  000002  114020          MOV     2(SP),@SWR      ;;SAVE NEW SWR
065134  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
065140  104401  001217          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
065144  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
065152  001003                          BNE     15$             ;;BRANCH IF NOT
065154  012777  000100  113776          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
065162  000002                  15$:    RTI                     ;;RETURN
065164  004737  061750          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
065170  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
065174  002420                          BLT     18$             ;;BRANCH IF YES
065176  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
065202  003015                          BGT     18$             ;;BRANCH IF YES
065204  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
065210  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
065214  001403                          BEQ     17$             ;;BRANCH IF YES
065216  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
065220  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
065222  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
065224  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
065230  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
065234  000667                          BR      7$              ;;GET THE NEXT ONE
065236  104401  001216          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
065242  000720                          BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB

                                ;;*****************************************************************
```

```
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;

065244  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
065246  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
065254  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
065260  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
065262  012746  065270                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
065266  000002                          RTI                     ;;POP NEW PC AND PS
065270                          64$:
065270  005737  064372          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
065274  001775                          BEQ     1$
065276  005337  064372                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
065302  117766  177070  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
065310  005237  064376                  INC     $TKQOUT         ;;UPDATE THE POINTER
065314  023727  064376  064401          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
065322  001003                          BNE     2$              ;;BRANCH IF NO
065324  012737  064400  064376          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
065332  000002                  2$:     RTI                     ;;RETURN
                                ;;**************************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

065334  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
065336  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
065340  012703  065570          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
065344  022703  065600          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
065350  101456                          BLOS    4$              ;;BR IF YES
065352  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
065354  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
065356  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
065362  001022                          BNE     5$              ;;BR IF NO
065364  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
065366  001007                          BNE     6$              ;;BR IF NO
065370  112737  000134  065566          MOVB    #'\,9$          ;;TYPE A BACK SLASH
065376  104401  065566                  TYPE    ,9$
065402  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
065406  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
065410  020327  065570                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
065414  103434                          BLO     4$              ;;BR IF YES
065416  111337  065566                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
065422  104401  065566                  TYPE    ,9$             ;;GO TYPE
065426  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
065430  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
065432  001406                          BEQ     7$              ;;BR IF NO
065434  112737  000134  065566          MOVB    #'\,9$          ;;TYPE A BACK SLASH
065442  104401  065566                  TYPE    ,9$
065446  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
065450  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
065454  001003                          BNE     8$              ;;BR IF NO
```

```
065456  104401  065605                  TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
065462  000726                          BR      1$              ;;GO START OVER
065464  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
065470  001011                          BNE     3$              ;;BRANCH IF NO
065472  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
065474  104401  001217                  TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
065500  104401  065570                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
065504  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
065506  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
065512  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
065514  111337  065566          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
065520  104401  065566                  TYPE    ,9$
065524  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
065530  001305                          BNE     2$              ;;LOOP IF NOT RETURN
065532  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
065536  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
065542  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
065544  012603                          MOV     (SP)+,R3        ;;RESTORE R3
065546  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
065550  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
065556  012766  065570  000004          MOV     #$TTYIN,4(SP)
065564  000002                          RTI                     ;;RETURN
065566     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
065567     000                  .BYTE   0                       ;;TERMINATOR
065570                  $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
065600     136     103     015  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
065605     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
065612     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
065617     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
065630     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
                                .EVEN
2
3 065642  005737  000042  SHUT:   TST     @#42            ;ANY MONITOR PRESENT ?
4 065646  001002                  BNE     1$              ;BR IF YES
5 065650  000137  005426          JMP     START           ;GO TO START
6 065654  005037  001300  1$:     CLR     $DEVM           ;FUDGE NO DRIVES IN MAP
7 065660  000137  041314          JMP     $EOP            ;RETURN TO $EOP
```

READ AN OCTAL NUMBER FROM THE TTY

```
     1                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                        ;;*************************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*CALL:
                                        ;*      RDOCT                       ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE                 ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                                  ;;HIGH ORDER BITS ARE IN $HIOCT

 065664  011646                 $RDOCT: MOV     (SP),-(SP)          ;;PROVIDE SPACE FOR THE
 065666  016666  000004  000002         MOV     4(SP),2(SP)         ;;INPUT NUMBER
 065674  010046                         MOV     R0,-(SP)            ;;PUSH R0 ON STACK
 065676  010146                         MOV     R1,-(SP)            ;;PUSH R1 ON STACK
 065700  010246                         MOV     R2,-(SP)            ;;PUSH R2 ON STACK
 065702  104412                 1$:     RDLIN                       ;;READ AN ASCIZ LINE
 065704  012600                         MOV     (SP)+,R0            ;;GET ADDRESS OF 1ST CHARACTER
 065706  005001                         CLR     R1                  ;;CLEAR DATA WORD
 065710  005002                         CLR     R2
 065712  112046                 2$:     MOVB    (R0)+,-(SP)         ;;PICKUP THIS CHARACTER
 065714  001412                         BEQ     3$                  ;;IF ZERO GET OUT
 065716  006301                         ASL     R1                  ;;*2
 065720  006102                         ROL     R2
 065722  006301                         ASL     R1                  ;;*4
 065724  006102                         ROL     R2
 065726  006301                         ASL     R1                  ;;*8
 065730  006102                         ROL     R2
 065732  042716  177770                BIC     #^C7,(SP)           ;;STRIP THE ASCII JUNK
 065736  062601                         ADD     (SP)+,R1            ;;ADD IN THIS DIGIT
 065740  000764                         BR      2$                  ;;LOOP
 065742  005726                 3$:     TST     (SP)+               ;;CLEAN TERMINATOR FROM STACK
 065744  010166  000012                MOV     R1,12(SP)           ;;SAVE THE RESULT
 065750  010237  065764                MOV     R2,$HIOCT
 065754  012602                         MOV     (SP)+,R2            ;;POP STACK INTO R2
 065756  012601                         MOV     (SP)+,R1            ;;POP STACK INTO R1
 065760  012600                         MOV     (SP)+,R0            ;;POP STACK INTO R0
 065762  000002                         RTI                         ;;RETURN
 065764  000000                 $HIOCT: .WORD   0                   ;;HIGH ORDER BITS GO HERE
```

```
                                .SBTTL  TRAP DECODER

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

065766  016646  000002        $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
065772  042716  000020                BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
065776  012746  066004                MOV     #1$,-(SP)       ;; T-BIT TRAPS
066002  000002                      RTI                     ;;SET THE NEW STATUS
066004  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
066006  016600  000002              MOV     2(SP),R0        ;;GET TRAP ADDRESS
066012  005740                      TST     -(R0)           ;;BACKUP BY 2
066014  111000                      MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
066016  006300                      ASL     R0              ;;POSITION FOR INDEXING
066020  016000  066040              MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
066024  000200                      RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

066026  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
066030  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
066036  000002                      RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
066040  066026                $TRPAD: .WORD   $TRAP2
066042  061536                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
066044  061334                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
066046  061310                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
066050  061350                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
066052  061064                        $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
066054  061010                        $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

066056  064772                        $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING

066060  064702                        $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
066062  065244                        $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
066064  065334                        $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
066066  065664                        $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
066070  060714                        $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
066072  060752                        $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
```

```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*************************************************************************
                                ;POWER DOWN ROUTINE
066074  012737  066234  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
066102  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
066110  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
066112  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
066114  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
066116  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
066120  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
066122  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
066124  017746  113024                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
066130  010637  066240                  MOV     SP,$SAVR6       ;;SAVE SP
066134  012737  066146  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
066142  000000                          HALT
066144  000776                          BR      .-2             ;;HANG UP

                                ;;*************************************************************************
                                ;POWER UP ROUTINE
066146  012737  066234  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
066154  013706  066240                  MOV     $SAVR6,SP       ;;GET SP
066160  005037  066240                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
066164  005237  066240          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
066170  001375                          BNE     1$              ;;OF  WORD
066172  012677  112756                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
066176  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
066200  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
066202  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
066204  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
066206  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
066210  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
066212  012737  066074  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
066220  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
066226  104401                          TYPE                    ;REPORT THE POWER FAILURE
066230  066242          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
066232  000002                  RTI
066234  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
066236  000776                  BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
066240  000000          $SAVR6: 0                               ;;PUT THE SP HERE
066242     015     012     120  $POWER: .ASCIZ   <15><12>'POWER'
                                .EVEN
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;**************************************************************************
066252  112737  000001  066516  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
066260  112737  000001  066514  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
066266  000403                          BR      $ATYC
066270  112737  000001  066516  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
066276                          $ATYC:
066276  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
066300  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
066302  105737  066514                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
066306  001450                          BEQ     5$              ;;IF NOT:  BR
066310  122737  000001  001242          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
066316  001031                          BNE     3$              ;;IF NOT:  BR
066320  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
066326  001425                          BEQ     3$              ;;IF NOT:  BR
066330  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
066334  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
066342  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
066346  001375                          BNE     1$              ;;IF NOT:  WAIT
066350  010037  001236                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
066354  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
066356  001376                          BNE     2$
066360  163700  001236                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
066364  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
066366  010037  001240                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
066372  012737  000004  001222          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
066400  000413                          BR      5$
066402  017637  000004  066426  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
066410  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
066416  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
066422  004737  061536                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
066426  000000                  4$:     .WORD   0
066430                          5$:
066430  105737  066516          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
066434  001416                          BEQ     12$             ;;IF NOT:  BR
066436  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
066442  001413                          BEQ     12$             ;;IF NOT:  BR
066444  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
066450  001375                          BNE     11$             ;;IF NOT:  WAIT
066452  017637  000004  001224          MOV     @4(SP),$FATAL   ;;GET ERROR #
066460  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
066466  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
066472  105037  066516          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
066476  105037  066515                  CLRB    $LFLG           ;;CLEAR LOG FLAG
066502  105037  066514                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
066506  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
066510  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
066512  000207                          RTS     PC              ;;RETURN
066514     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
066515     000          $LFLG:  .BYTE   0               ;;LOG FLAG
066516     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
                        APTSIZE = 200
                        APTENV  = 001
                        APTSPOOL= 100
                        APTCSUP = 040
```

```
                                            .SBTTL  CONSOLE MESSAGES

066520   075    000             EQUALS:  .ASCIZ   @=@
066522   101    114    114      ALL:     .ASCIZ   @ALL@<CRLF>
066527   040    077    040      QUES:    .ASCIZ   @ ? @
066533   054    040    000      COMMA:   .ASCIZ   @, @
066536   200    124    131      MSHELP:  .ASCIZ   <CRLF>@TYPE HELP TEXT (L) N ? @
066567   200    122    115      CNSL01:  .ASCIZ   <CRLF>@RMCS1=@
066577   040    114    111      CNSL02:  .ASCIZ   @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
066641   122    115    126      CNSL03:  .ASCIZ   @RMVEC=@
066650   040    114    111      CNSL04:  .ASCIZ   @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
066704   200    124    131      CNSL07:  .ASCII   <CRLF>@TYPE "A" TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
066771   200    101    116               .ASCIZ   <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
067046   200                    CNSL08:  .ASCII   <CRLF>
067047   040    077    111      CNSL09:  .ASCIZ   @ ?ILLEGAL INPUT@<CRLF>
067070   200    104    122      MSDRVS:  .ASCIZ   <CRLF>/DRIVE(S): /
067104   104    122    111      MSGDRV:  .ASCIZ   /DRIVE/
067112   200    125    116      SYSTAT:  .ASCIZ   <CRLF>/UNIT STATUS:/
067130   122    115    070      $RM80:   .ASCIZ   /RM80/
067135   040    116    117      NOTRM:   .ASCIZ   / NOT AN RM80/
067152   040    114    117      LODEV:   .ASCIZ   / LOAD DEVICE/
067167   040    116    117      NOTPRS:  .ASCIZ   / NOT PRESENT/
067204   040    116    117      NOTAVL:  .ASCIZ   / NOT AVAILABLE/
067223   040    117    106      UNTOFF:  .ASCIZ   / OFFLINE/
067234   040    117    116      UNTON:   .ASCIZ   / ONLINE/
067244   200    104    122      DRIVES:  .ASCIZ   <CRLF>/DRIVE(S) TO BE TESTED/
067273   116    117    116      NONE:    .ASCIZ   /NONE/
067300   200    104    117      MSMNUL:  .ASCIZ   <CRLF>/DO YOU WANT MANUAL INTERVENTION TESTS (L) N ? /
067360   200    120    105      MSUNIT:  .ASCIZ   <CRLF>/PERFORMING 'LOGICAL ADDRESS PLUG TEST' ON DRIVE #/
067443   200    012    102      MINSTR:  .ASCII   <CRLF><LF>/BEFORE PROCEDING WITH THE TEST, POWER DOWN OR PULL/
067527   200    124    110               .ASCIZ   <CRLF>/THE ADDRESS PLUGS FROM ALL DEVICES NOT BEING TESTED./<CRLF><LF>
067617   111    116    123      INSERT:  .ASCIZ   /INSERT LOGICAL ADDRESS PLUG #/
067655   120    125    114      PULL:    .ASCIZ   /PULL LOGICAL ADDRESS PLUG #0, THEN/
067720   200    124    131      RET:     .ASCIZ   <CRLF>/TYPE <RET> WHEN READY./
067750   120    114    125      DONE:    .ASCIZ   /PLUG TEST DONE!/<CRLF>
067771   116    000             N:       .ASCIZ   /N/
067773   131    000             Y:       .ASCIZ   /Y/
067775   040                    BLNKS4:  .ASCII   / /
067776   040                    BLNKS3:  .ASCII   / /
067777   040                    BLNKS2:  .ASCII   / /
070000   040    000             BLNKS1:  .ASCIZ   / /
                                         .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", "BSE" AND "SSE".

                        ;       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.

070002                  FNCDTB:                                 ;FUNCTION CODE TABLE

070002  020000          .WORD   OPI                             ;NOP
```

FUNCTION CODE TABLE

```
     58 070004  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
     59 070006  132000                .WORD   ATA!OPI!IVC!IAE                         ;SEEK
     60 070010  130000                .WORD   ATA!OPI!IVC                             ;RECALIBRATE
     61 070012  020000                .WORD   OPI                                     ;DRIVE CLEAR
     62 070014  030000                .WORD   OPI!IVC                                 ;RELEASE
     63 070016  130000                .WORD   OPI!ATA!IVC                             ;OFFSET
     64 070020  130000                .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
     65 070022  020000                .WORD   OPI                                     ;READ IN PRESET
     66 070024  020000                .WORD   OPI                                     ;PACK ACKNOWLEDGE
     67 070026  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
     68 070030  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
     69 070032  132000                .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
     70 070034  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
     71 070036  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
     72 070040  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
     73 070042  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
     74 070044  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
     75 070046  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
     76 070050  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
     77 070052  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
     78 070054  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
     79 070056  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
     80 070060  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
     81 070062  037200                .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
     82 070064  037000                .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
     83 070066  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
     84 070070  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
     85 070072  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
     86 070074  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
     87 070076  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
     88 070100  130001                .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                                .SBTTL  ATTENTION (ATA) TABLE
                        
070102      001         ATNTBL: .BYTE   1.
070103      002                 .BYTE   2.
070104      004                 .BYTE   4.
070105      010                 .BYTE   8.
070106      020                 .BYTE   16.
070107      040                 .BYTE   32.
070110      100                 .BYTE   64.
070111      200                 .BYTE   128.
```

ERROR MESSAGE TABLE

```
                                .SBTTL  ERROR MESSAGE TABLE
                        
070112  074564  000000  EMT1:   .WORD   EMS1,0
070116  074633  074650  000000  EMT2:   .WORD   EMS2,EMS3,0
070124  074633  074713  000000  EMT3:   .WORD   EMS2,EMS4,0
070132  074756  075006  000000  EMT4:   .WORD   EMS5,EMS6,0
070140  074756  075120  000000  EMT5:   .WORD   EMS5,EMS10,0
070146  101552  077001  000000  EMT6:   .WORD   EMS167,EMS64,0
070154  077547  101577  000000  EMT7:   .WORD   EMS110,EMS170,0
070162  075053  000000          EMT10:  .WORD   EMS7,0
070166  075120  000000          EMT11:  .WORD   EMS10,0
070172  075162  075173  000000  EMT12:  .WORD   EMS11,EMS12,0
070200  075234  075245  075256  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
070212  075330  077001  000000  EMT14:  .WORD   EMS17,EMS64,0
070220  075162  075413  000000  EMT15:  .WORD   EMS11,EMS21,0
070226  075162  075436  075565  EMT16:  .WORD   EMS11,EMS22,EMS27,0
070236  075162  075452  075576  EMT17:  .WORD   EMS11,EMS23,EMS30,0
070246  075162  075500  075576  EMT20:  .WORD   EMS11,EMS24,EMS30,0
070256  075162  075527  075565  EMT21:  .WORD   EMS11,EMS25,EMS27,0
070266  075162  075544  075565  EMT22:  .WORD   EMS11,EMS26,EMS27,0
070276  075162  075606  075576  EMT23:  .WORD   EMS11,EMS31,EMS30,0
070306  075162  075635  075576  EMT24:  .WORD   EMS11,EMS32,EMS30,0
070316  075162  075664  075576  EMT25:  .WORD   EMS11,EMS33,EMS30,0
070326  075162  075712  075576  EMT26:  .WORD   EMS11,EMS34,EMS30,0
070336  075162  075763  075576  EMT27:  .WORD   EMS11,EMS35,EMS30,0
070346  075162  076012  075576  EMT30:  .WORD   EMS11,EMS36,EMS30,0
070356  075162  076041  075576  EMT31:  .WORD   EMS11,EMS37,EMS30,0
070366  075162  076067  075576  EMT32:  .WORD   EMS11,EMS40,EMS30,0
070376  075162  076116  075576  EMT33:  .WORD   EMS11,EMS41,EMS30,0
070406  075162  076144  075576  EMT34:  .WORD   EMS11,EMS42,EMS30,0
070416  075162  076173  075576  EMT35:  .WORD   EMS11,EMS43,EMS30,0
070426  075162  076222  075576  EMT36:  .WORD   EMS11,EMS44,EMS30,0
070436  075162  076275  075576  EMT37:  .WORD   EMS11,EMS45,EMS30,0
070446  077065  075370  000000  EMT40:  .WORD   EMS66,EMS20,0
070454  077253  100753  077261  EMT41:  .WORD   EMS75,EMS141,EMS76,0
070464  101044  101054  077207  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
070476  076367  076505  077261  EMT43:  .WORD   EMS47,EMS53,EMS76,0
070506  077312  076505  077261  EMT44:  .WORD   EMS77,EMS53,EMS76,0
070516  077340  076505  077261  EMT45:  .WORD   EMS100,EMS53,EMS76,0
070526  077366  076505  077261  EMT46:  .WORD   EMS101,EMS53,EMS76,0
070536  077017  077001  000000  EMT47:  .WORD   EMS65,EMS64,0
070544  075234  075256  076753  EMT50:  .WORD   EMS13,EMS15,EMS63,0
070554  075162  076340  075576  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
070566  076367  076505  077135  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
070604  076367  076505  077135  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
070622  076416  076505  077135  EMT54:  .WORD   EMS50,EMS53,EMS67,0
070632  101001  101016  076505  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
070644  076445  077207  077135  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
070662  101552  077217  077135  EMT57:  .WORD   EMS167,EMS73,EMS67,0
070672  076570  077135  077747  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
070710  077174  076570  077135  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
070730  100732  077135  077747  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
070744  102752  102775  075370  EMT63:  .WORD   EMS225,EMS226,EMS20,0
070754  101137  101202  077162  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
070774  076473  102177  102360  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
071014  101511  077443  077207  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
071026  101511  077443  077207  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
```

```
     58 071040  076340  075370  101404  EMT70:  .WORD   EMS46,EMS20,EMS163,0
     59 071050  077174  077366  101404  EMT71:  .WORD   EMS71,EMS101,EMS163,0
     60 071060  076367  077207  101404  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
     61 071076  076367  077207  101404  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
     62 071114  076570  076505  101404  EMT74:  .WORD   EMS56,EMS53,EMS163,0
     63 071124  077174  076570  101404  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
     64 071144  076416  076505  101404  EMT76:  .WORD   EMS50,EMS53,EMS163,0
     65 071154  101001  101016  076505  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
     66 071166  077253  100753  101404  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
     67 071204  077253  101137  101404  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
     68 071222  101552  077217  101404  EMT102: .WORD   EMS167,EMS73,EMS163,0
     69 071232  101122  101156  077234  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
     70 071244  076445  077234  101404  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
     71 071262  101511  077340  101404  EMT105: .WORD   EMS165,EMS100,EMS163,0
     72 071272  101511  077312  101404  EMT106: .WORD   EMS165,EMS77,EMS163,0
     73 071302  101044  101054  076505  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
     74 071322  077547  077607  077752  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
     75 071334  077673  074713  000000  EMT111: .WORD   EMS113,EMS4,0
     76 071342  077673  074650  000000  EMT112: .WORD   EMS113,EMS3,0
     77 071350  077547  077747  077752  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
     78 071364          EMT114:         ;NOT USED
     79 071364  100024  100464  100510  EMT115: .WORD   EMS121,EMS132,EMS133,0
     80 071374  100067  100464  100510  EMT116: .WORD   EMS122,EMS132,EMS133,0
     81 071404  100124  100464  100510  EMT117: .WORD   EMS123,EMS132,EMS133,0
     82 071414  100167  100464  100510  EMT120: .WORD   EMS124,EMS132,EMS133,0
     83 071424  100221  100464  100510  EMT121: .WORD   EMS125,EMS132,EMS133,0
     84 071434  100264  100464  100510  EMT122: .WORD   EMS126,EMS132,EMS133,0
     85 071444  100675  100464  100510  EMT123: .WORD   EMS137,EMS132,EMS133,0
     86 071454  100366  100464  100510  EMT124: .WORD   EMS130,EMS132,EMS133,0
     87 071464  100424  100464  100510  EMT125: .WORD   EMS131,EMS132,EMS133,0
     88 071474  100024  100464  100533  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
     89 071506  100067  100464  100533  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
     90 071520  100124  100464  100533  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
     91 071532  100167  100464  100533  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
     92 071544  100221  100464  100533  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
     93 071556  100264  100464  100533  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
     94 071570  100675  100464  100533  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
     95 071602  100366  100464  100533  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
     96 071614  100424  100464  100533  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
     97 071626  100024  100464  100575  EMT137: .WORD   EMS121,EMS132,EMS135,0
     98 071636  100124  100464  100575  EMT140: .WORD   EMS123,EMS132,EMS135,0
     99 071646  100024  100464  100650  EMT141: .WORD   EMS121,EMS132,EMS136,0
    100 071656  100675  100464  100650  EMT142: .WORD   EMS137,EMS132,EMS136,0
    101 071666  100167  100464  100650  EMT143: .WORD   EMS124,EMS132,EMS136,0
    102 071676  100221  100464  100650  EMT144: .WORD   EMS125,EMS132,EMS136,0
    103 071706  100264  100464  100650  EMT145: .WORD   EMS126,EMS132,EMS136,0
    104 071716  100424  100464  100650  EMT146: .WORD   EMS131,EMS132,EMS136,0
    105 071726  101626  100464  100650  EMT147: .WORD   EMS171,EMS132,EMS136,0
    106 071736  100366  100464  100650  EMT150: .WORD   EMS130,EMS132,EMS136,0
    107 071746  100732  077747  100753  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
    108 071760  101001  077747  101016  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
    109 071772  101044  101054  101103  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
    110 072010  101044  101054  075370  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
    111 072026  101137  101202  101250  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
    112 072040  101122  101156  101250  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
    113 072052  101122  101156  101304  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
    114 072064  101354  101304  101337  EMT160: .WORD   EMS161,EMS156,EMS160,0
```

```
115 072074  075527  075565  101304  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
116 072106  075544  075565  101304  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
117 072120  076367  101404  100732  EMT163: .WORD   EMS47,EMS163,EMS140,0
118 072130  076367  075370  000000  EMT164: .WORD   EMS47,EMS20,0
119 072136  076570  075370  000000  EMT165: .WORD   EMS56,EMS20,0
120 072144  076340  075370  000000  EMT166: .WORD   EMS46,EMS20,0
121 072152  102700  075370  000000  EMT167: .WORD   EMS224,EMS20,0
122 072160  102136  100753  075576  EMT170: .WORD   EMS203,EMS141,EMS30,EMS202,0
123 072172  101552  101250  101322  EMT171: .WORD   EMS167,EMS154,EMS157,0
124 072202  101552  101250  101264  EMT172: .WORD   EMS167,EMS154,EMS155,0
125 072212  101626  100464  100510  EMT173: .WORD   EMS171,EMS132,EMS133,0
126 072222  101626  100464  100533  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
127 072234  077673  102001  000000  EMT175: .WORD   EMS113,EMS177,0
128 072242  102023  102040  000000  EMT176: .WORD   EMS200,EMS201,0
129 072250  000000                  EMT177: .WORD
130 072252  102136  076445  075576  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
131 072264  102136  102151  075576  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
132 072276  102136  076416  075576  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
133 072310  102136  101016  075576  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
134 072322  101137  077234  102106  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
135 072340  076416  077443  077207  EMT205: .WORD   EMS50,EMS103,EMS72,0
136 072350  077452  077217  102106  EMT206: .WORD   EMS104,EMS73,EMS202,0
137 072360  077253  100753  077747  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
138 072372  077253  101137  000000  EMT210: .WORD   EMS75,EMS150,0
139 072400  076445  077207  077747  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
140 072414  101001  076505  077747  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
141 072430  076416  077443  076505  EMT213: .WORD   EMS50,EMS103,EMS53,0
142 072440  076473  102177  101525  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
143 072460  076473  100002  076570  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
144 072472  076570  075576  077001  EMT216: .WORD   EMS56,EMS30,EMS64,0
145 072502  076340  075576  077001  EMT217: .WORD   EMS46,EMS30,EMS64,0
146 072512  075606  075576  077001  EMT220: .WORD   EMS31,EMS30,EMS64,0
147 072522  076367  075576  077001  EMT221: .WORD   EMS47,EMS30,EMS64,0
148 072532  076473  100002  077312  EMT222: .WORD   EMS52,EMS117,EMS77,0
149 072542  076473  100002  076753  EMT223: .WORD   EMS52,EMS117,EMS63,0
150 072552  076473  100002  102752  EMT224: .WORD   EMS52,EMS117,EMS225,EMS115,EMS226,0
151 072566  102752  103024  103035  EMT225: .WORD   EMS225,EMS227,EMS230,EMS115,EMS156,EMS231,EMS57,0
152 072606  075162  103035  075576  EMT226: .WORD   EMS11,EMS230,EMS30,0
153 072616  102752  102775  077747  EMT227: .WORD   EMS225,EMS226,EMS115,EMS230,EMS72,0
154 072632  074564  000000          EMT230: .WORD   EMS1,0
155 072636  102136  101752  075576  EMT231: .WORD   EMS203,EMS176,EMS30,0
156 072646  000000                  EMT232: .WORD
157 072650  000000                  EMT233: .WORD
158 072652  000000                  EMT234: .WORD
159 072654  000000                  EMT235: .WORD
160 072656  000000                  EMT236: .WORD
161 072660  000000                  EMT237: .WORD
162 072662  000000                  EMT240: .WORD
163 072664  000000                  EMT241: .WORD
164 072666  000000                  EMT242: .WORD
165 072670  000000                  EMT243: .WORD
166 072672  000000                  EMT244: .WORD
167 072674  000000                  EMT245: .WORD
168 072676  101552  100464  102236  EMT246: .WORD   EMS167,EMS132,EMS207,0
169 072706  101552  100464  102263  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
170 072720  101552  102274  102263  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
171 072734  102312  102335  000000  EMT251: .WORD   EMS212,EMS213,0
```

```
172 072742  101511  102312  000000  EMT252: .WORD   EMS165,EMS212,0
173 072750  101122  101156  101304  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
174 072766  102136  077366  075576  EMT254: .WORD   EMS203,EMS101,EMS30,0
175 072776  102136  101552  075576  EMT255: .WORD   EMS203,EMS167,EMS30,0
176 073006  102136  077340  075576  EMT256: .WORD   EMS203,EMS100,EMS30,0
177 073016  075162  076340  075576  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
178 073030  076473  100002  076570  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
179 073042  076473  102177  102502  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
180 073062  077452  077217  102106  EMT262: .WORD   EMS104,EMS73,EMS202,0
181 073072  076416  076505  077414  EMT263: .WORD   EMS50,EMS53,EMS102,0
182 073102  076570  076505  077414  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
183 073122  077174  076570  077414  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
184 073142  101001  101016  076505  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
185 073154  076416  101103  077747  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
186 073172  076367  076505  077414  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
187 073206  076367  076505  077414  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
188 073224  077253  100753  077414  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
189 073242  076445  077234  077414  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
190 073260  076753  076505  076645  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
191 073276  077752  076505  076116  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
192 073310  076367  076505  076645  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
193 073324  076367  076505  076645  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
194 073342  076570  076505  076645  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
195 073362  077174  076570  076505  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
196 073404  101511  077340  077443  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
197 073416  101511  077366  077443  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
198 073430  101511  077312  077443  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
199 073442  076340  075576  077001  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
200 073454  076416  076505  076645  EMT306: .WORD   EMS50,EMS53,EMS57,0
201 073464  076416  101103  077747  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
202 073504  101001  101016  076505  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
203 073516  077452  077443  076505  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
204 073530  077501  077443  076505  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
205 073542  101044  101054  077443  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
206 073562  076144  077443  076505  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
207 073574  075606  077443  076505  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
208 073606  077174  075606  077443  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
209 073620  076173  077443  076645  EMT317: .WORD   EMS43,EMS103,EMS57,0
210 073630  076275  077443  076645  EMT320: .WORD   EMS45,EMS103,EMS57,0
211 073640  076222  077443  076645  EMT321: .WORD   EMS44,EMS103,EMS57,0
212 073650  077530  075370  000000  EMT322: .WORD   EMS106,EMS20,0
213 073656  076012  077443  076645  EMT323: .WORD   EMS36,EMS103,EMS57,0
214 073666  101701  076012  077443  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
215 073700  101661  076012  077443  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
216 073712  075234  101716  075256  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
217 073730  101122  101156  077234  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
218 073742  077065  076505  101724  EMT330: .WORD   EMS66,EMS53,EMS175,0
219 073752  075664  077443  076505  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
220 073764  076067  077443  076505  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
221 073776  076445  077234  076645  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
222 074014  077253  100753  076645  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
223 074032  077253  101137  101202  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
224 074052  076673  076706  076735  EMT336: .WORD   EMS60,EMS61,EMS62,0
225 074062  077752  100002  075712  EMT337: .WORD   EMS116,EMS117,EMS34,0
226 074072  075712  076505  076517  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
227 074104  076541  075712  000000  EMT341: .WORD   EMS55,EMS34,0
228 074112  076473  100002  076570  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
```

```
    229 074124  076473  100002  076275  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
    230 074136  076473  100002  076222  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
    231 074150  076473  100002  102522  EMT345: .WORD   EMS52,EMS117,EMS221,0
    232 074160  077530  075370  077747  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
    233 074174  076473  102177  102572  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
    234 074206  077253  101137  077414  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
    235 074224  101552  077217  077414  EMT351: .WORD   EMS167,EMS73,EMS102,0
    236 074234  102376  000000          EMT352: .WORD   EMS215,0
    237 074240  102447  102136  102417  EMT353: .WORD   EMS217,EMS203,EMS216,0
    238 074250  102522  075370  000000  EMT354: .WORD   EMS221,EMS20,0
```

ERROR MESSAGE TABLE

```
 1 074256  103127  103725  104002  EHT1:    .WORD   EH1,STSH1,STSH2,STSH4,0
 2 074270  103725  104002  104127  EHT2:    .WORD   STSH1,STSH2,STSH4,0
 3
 4 074300  103146  000000          EHT110:  .WORD   EH110,0
 5 074304  103155  000000          EHT111:  .WORD   EH111,0
 6 074310  103127  000000          EHT112:  .WORD   EH1,0
 7 074314  103174  000000          EHT114:  .WORD   EH114,0
 8 074320  103223  103725  104002  EHT223:  .WORD   EH223,STSH1,STSH2,STSH4,0
 9 074332  103251  103725  104002  EHT256:  .WORD   EH256,STSH1,STSH2,STSH4,0
10
11 074344  103325  103725  104002  EHT336:  .WORD   EH336,STSH1,STSH2,STSH4,0
12 074356  103364  103725  104002  EHT337:  .WORD   EH337,STSH1,STSH2,STSH4,0
13 074370  103521  103725  104002  EHT344:  .WORD   EH344,STSH1,STSH2,STSH4,0
14
15 074402  103657  000000          EHT353:  .WORD   EH353,0
```

ERROR MESSAGE TABLE

```
      1 074406  104166  104262  104300  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
      2 074416  104262  104300  104332  EDT2:    .WORD   STSD1,STSD2,STSD4
      3
      4 074424  104174                  EDT110:  .WORD   ED110
      5 074426  104200                  EDT111:  .WORD   ED111
      6
      7 074430  104206                  EDT114:  .WORD   ED114
      8 074432  104216  104262  104300  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4
      9
     10 074442  104226  104262  104300  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
     11 074452  104240  104262  104300  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
     12 074462  104240  104262  104300  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0
     13
     14 074474  104252                  EDT353:  .WORD   ED353
```

ERROR MESSAGE TABLE

```
 1 074476  104345  104363  104363  EFT1:    .WORD   EF111,STSF,STSF,STSF
 2 074506  104363  104363  104363  EFT2:    .WORD   STSF,STSF,STSF
 3
 4 074514  104344                  EFT110:  .WORD   EF110
 5 074516  104345                  EFT111:  .WORD   EF111
 6
 7 074520  104347                  EFT114:  .WORD   EF114
 8 074522  104347  104363  104363  EFT223:  .WORD   EF114,STSF,STSF,STSF
 9
10 074532  104352  104363  104363  EFT336:  .WORD   EF336,STSF,STSF,STSF
11 074542  104352  104363  104363  EFT337:  .WORD   EF336,STSF,STSF,STSF
12 074552  104352  104363  104363  EFT344:  .WORD   EF336,STSF,STSF,STSF
13
14 074562  104347                  EFT353:  .WORD   EF114
```

```
                                        .SBTTL  ERROR MESSAGE STRINGS

074564   127   122   117   EMS1:    .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
074633   104   105   126   EMS2:    .ASCIZ  @DEVICE WENT @
074650   125   116   101   EMS3:    .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
074713   116   117   116   EMS4:    .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
074756   103   117   115   EMS5:    .ASCIZ  @COMMAND NOT COMPLETED, @
075006   103   117   116   EMS6:    .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
075053   104   122   111   EMS7:    .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
075120   107   117   040   EMS10:   .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
075162   111   116   126   EMS11:   .ASCIZ  @INVALID @
075173   106   125   116   EMS12:   .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
075234   115   101   123   EMS13:   .ASCIZ  @MASSBUS @
075245   103   117   116   EMS14:   .ASCIZ  @CONTROL @
075256   102   125   123   EMS15:   .ASCIZ  @BUS PARITY ERROR @
075300   042   115   103   EMS16:   .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
075330   124   122   101   EMS17:   .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
075370   123   110   117   EMS20:   .ASCIZ  @SHOULD NOT BE SET @
075413   127   117   122   EMS21:   .ASCIZ  @WORD COUNT (RMWC) @
075436   102   125   123   EMS22:   .ASCIZ  @BUS (RMBA) @
075452   042   114   102   EMS23:   .ASCIZ  @"LBT" (RMDS, BIT 10) @
075500   042   101   117   EMS24:   .ASCIZ  @"AOE" (RMER1, BIT 09) @
075527   104   111   123   EMS25:   .ASCIZ  @DISK (RMDA) @
075544   103   131   114   EMS26:   .ASCIZ  @CYLINDER (RMDC) @
075565   101   104   104   EMS27:   .ASCIZ  @ADDRESS @
075576   123   124   101   EMS30:   .ASCIZ  @STATUS @
075606   042   127   114   EMS31:   .ASCIZ  @"WLE" (RMER1, BIT 11) @
075635   042   125   120   EMS32:   .ASCIZ  @"UPE" (RMCS2, BIT 13) @
075664   042   127   103   EMS33:   .ASCIZ  @"WCF" (RMER1, BIT 5) @
075712   127   122   111   EMS34:   .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
075763   042   115   104   EMS35:   .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
076012   042   104   103   EMS36:   .ASCIZ  @"DCK" (RMER1, BIT 15) @
076041   042   105   103   EMS37:   .ASCIZ  @"ECH" (RMER1, BIT 6) @
076067   042   104   114   EMS40:   .ASCIZ  @"DLT" (RMCS2, BIT 15) @
076116   042   115   130   EMS41:   .ASCIZ  @"MXF" (RMCS2, BIT 9) @
076144   042   104   124   EMS42:   .ASCIZ  @"DTE" (RMER1, BIT 12) @
076173   042   110   103   EMS43:   .ASCIZ  @"HCRC" (RMER1, BIT 8) @
076222   110   105   101   EMS44:   .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
076275   106   117   122   EMS45:   .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
076340   042   111   101   EMS46:   .ASCIZ  @"IAE" (RMER1, BIT 10) @
076367   042   117   120   EMS47:   .ASCIZ  @"OPI" (RMER1, BIT 13) @
076416   042   123   113   EMS50:   .ASCIZ  @"SKI" (RMER2, BIT 14) @
076445   042   120   111   EMS51:   .ASCIZ  @"PIP" (RMDS, BIT 13) @
076473   124   110   105   EMS52:   .ASCIZ  @THE RM80 @
076505   104   105   124   EMS53:   .ASCIZ  @DETECTED @
076517   101   124   040   EMS54:   .ASCIZ  @AT AN UNEXPECTED @
076541   111   116   103   EMS55:   .ASCIZ  @INCORRECT DATA DURING @
076570   111   116   126   EMS56:   .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
076645   104   125   122   EMS57:   .ASCIZ  @DURING DATA TRANSFER @
076673   104   101   124   EMS60:   .ASCIZ  @DATA READ @
076706   104   117   105   EMS61:   .ASCIZ  @DOES NOT COMPARE WITH @
076735   104   101   124   EMS62:   .ASCIZ  @DATA WRITTEN @
076753   042   120   101   EMS63:   .ASCIZ  @"PAR" (RMER1, BIT 3) @
077001   111   123   040   EMS64:   .ASCIZ  @IS INCORRECT @
077017   103   117   115   EMS65:   .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
077065   104   101   124   EMS66:   .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
077135   104   125   122   EMS67:   .ASCIZ  @DURING SEEK COMMAND @
```

ERROR MESSAGE STRINGS

```
     58 077162      111      123      040  EMS70:   .ASCIZ  @IS RESET @
     59 077174      105      122      122  EMS71:   .ASCIZ  @ERRONEOUS @
     60 077207      111      123      040  EMS72:   .ASCIZ  @IS SET @
     61 077217      104      111      104  EMS73:   .ASCIZ  @DID NOT SET @
     62 077234      104      111      104  EMS74:   .ASCIZ  @DID NOT RESET @
     63 077253      114      117      123  EMS75:   .ASCIZ  @LOST @
     64 077261      104      125      122  EMS76:   .ASCIZ  @DURING PACK ACK COMMAND @
     65 077312      042      122      115  EMS77:   .ASCIZ  @"RMR" (RMER1, BIT 2) @
     66 077340      042      111      114  EMS100:  .ASCIZ  @"ILR" (RMER1, BIT 1) @
     67 077366      042      111      114  EMS101:  .ASCIZ  @"ILF" (RMER1, BIT 0) @
     68 077414      104      125      122  EMS102:  .ASCIZ  @DURING SEARCH COMMAND @
     69 077443      105      122      122  EMS103:  .ASCIZ  @ERROR @
     70 077452      042      114      102  EMS104:  .ASCIZ  @"LBC" (RMER2, BIT 10) @
     71 077501      042      114      123  EMS105:  .ASCIZ  @"LSC" (RMER2, BIT 11) @
     72 077530      110      105      101  EMS106:  .ASCIZ  @HEADER ERRORS @
     73 077547      102      125      123  EMS110:  .ASCIZ  @BUS TIMEOUT (04 TRAP) @
     74 077576      101      104      104  EMS111:  .ASCIZ  @ADDRESS @
     75 077607      127      110      105  EMS112:  .ASCII  @WHEN READING/WRITING RH REGISTERS @
     76 077651      101      124      040           .ASCIZ  @AT THE FOLLOWING @
     77 077673      124      110      105  EMS113:  .ASCIZ  @THE SELECTED DEVICE IS @
     78 077723      116      117      116  EMS114:  .ASCIZ  @NONEXISTENT DEVICE @
     79 077747      015      012      000  EMS115:  .ASCIZ  <CR><LF>
     80 077752      124      110      105  EMS116:  .ASCIZ  @THE MASSBUS CONTROLLER @
     81 100002      106      101      111  EMS117:  .ASCIZ  @FAILED TO DETECT @
     82 100024                          EMS120:           ;NOT USED
     83 100024      103      117      116  EMS121:  .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
     84 100067      102      125      123  EMS122:  .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
     85 100124      103      117      116  EMS123:  .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
     86 100167      105      122      122  EMS124:  .ASCIZ  @ERROR REGISTER 1, RMER1, @
     87 100221      101      124      124  EMS125:  .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
     88 100264      115      101      111  EMS126:  .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
     89 100327      105      103      103  EMS127:  .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
     90 100366      105      103      103  EMS130:  .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
     91 100424      115      101      111  EMS131:  .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
     92 100464      116      117      124  EMS132:  .ASCIZ  @NOT INITIALIZED BY @
     93 100510      125      116      111  EMS133:  .ASCIZ  @UNIBUS INITIALIZE @
     94 100533      103      117      116  EMS134:  .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
     95 100575      122      110      040  EMS135:  .ASCIZ  @RH CONTROLLER ERROR CLEAR (RMCS1, BIT 14) @
     96 100650      104      122      111  EMS136:  .ASCIZ  @DRIVE CLEAR COMMAND @
     97 100675      104      122      111  EMS137:  .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
     98 100732      115      105      104  EMS140:  .ASCIZ  @MEDIUM OFF LINE @
     99 100753      042      115      117  EMS141:  .ASCIZ  @"MOL" (RMDS, BIT 12) @
    100 101001      104      122      111  EMS142:  .ASCIZ  @DRIVE FAULT @
    101 101016      042      104      126  EMS143:  .ASCIZ  @"DVC" (RMER2, BIT 7) @
    102 101044      125      116      123  EMS144:  .ASCIZ  @UNSAFE @
    103 101054      042      125      116  EMS145:  .ASCIZ  @"UNS" (RMER1, BIT 14) @
    104 101103      123      110      117  EMS146:  .ASCIZ  @SHOULD BE SET @
    105 101122      117      106      106  EMS147:  .ASCIZ  @OFFSET MODE @
    106 101137      040      126      117  EMS150:  .ASCIZ   @ VOLUME VALID @
    107 101156      042      117      115  EMS151:  .ASCIZ  @"OM" (RMDS, BIT 0) @
    108 101202      042      126      126  EMS152:  .ASCIZ  @"VV" (RMDS, BIT 6) @
    109 101226      120      101      103  EMS153:  .ASCIZ  @PACK ACK COMMAND @
    110 101250      116      117      124  EMS154:  .ASCIZ  @NOT SET BY @
    111 101264      117      106      106  EMS155:  .ASCIZ  @OFFSET COMMAND @
    112 101304      116      117      124  EMS156:  .ASCIZ  @NOT RESET BY @
    113 101322      122      124      103  EMS157:  .ASCIZ  @RTC COMMAND @
    114 101337      122      111      120  EMS160:  .ASCIZ  @RIP COMMAND @
```

```
    115 101354      117     106     106  EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
    116 101404                           EMS162:         ;<UNUSED>
    117 101404      104     125     122  EMS163: .ASCIZ  @DURING RECALIBRATE @
    118 101430      111     123     040  EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
    119 101477      103     131     114          .ASCIZ  @CYLINDER @
    120 101511      125     116     105  EMS165: .ASCIZ  @UNEXPECTED @
    121 101525      122     105     103  EMS166: .ASCIZ  @RECALIBRATE COMMAND @
    122 101552      042     101     124  EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
    123 101577      127     110     105  EMS170: .ASCIZ  @WHEN READING REGISTER @
    124 101626      105     122     122  EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
    125 101661      116     117     116  EMS172: .ASCIZ  @NONRECOVERABLE @
    126 101701      122     105     103  EMS173: .ASCIZ  @RECOVERABLE @
    127 101716      104     101     124  EMS174: .ASCIZ  @DATA @
    128 101724      104     125     122  EMS175: .ASCIZ  @DURING WRITE COMMAND @
    129 101752      042     117     120  EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
    130 102001      111     116     040  EMS177: .ASCIZ  @IN WRITE PROTECT @
    131 102023      103     101     116  EMS200: .ASCIZ  @CAN NOT SET @
    132 102040      104     111     101  EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
    133 102106      104     125     122  EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
    134 102136      111     116     103  EMS203: .ASCIZ  @INCORRECT @
    135 102151      042     127     122  EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
    136 102177      105     130     105  EMS205: .ASCIZ  @EXECUTED @
    137 102211      127     111     124  EMS206: .ASCIZ  @WITH COMP ERROR SET @
    138 102236      042     107     117  EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
    139 102263      127     122     111  EMS210: .ASCIZ  @WRITING @
    140 102274      127     101     123  EMS211: .ASCIZ  @WAS RESET BY @
    141 102312      120     122     117  EMS212: .ASCIZ  @PROGRAM INTERRUPT @
    142 102335      127     101     123  EMS213: .ASCIZ  @WAS NOT GENERATED @
    143 102360      123     105     105  EMS214: .ASCIZ  @SEEK COMMAND @
    144 102376      120     122     117  EMS215: .ASCIZ  @PROGRAM TIMEOUT @
    145 102417      104     125     122  EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
    146 102447      114     117     117  EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
    147 102502      123     105     101  EMS220: .ASCIZ  @SEARCH COMMAND @
    148 102522      102     101     104  EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
    149 102572      101     040     104  EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
    150 102623      110     105     101  EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
    151 102700      116     117     116  EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
    152 102752      123     113     111  EMS225: .ASCIZ  @SKIP SECTOR ERROR @
    153 102775      042     123     123  EMS226: .ASCIZ  @"SSE" (RMER2, BIT 05) @
    154 103024      111     116     110  EMS227: .ASCIZ  @INHIBIT @
    155 103035      042     123     123  EMS230: .ASCIZ  @"SSEI" (RMOF, BIT 09) @
    156 103064      105     116     104  EMS231: .ASCIZ  @END OF TRACK @
    157 103102      101     106     124  EMS232: .ASCIZ  @AFTER DATA TRANSFER @
```

ERROR MESSAGE STRINGS

```
     1 103127      105     130     120  EH1:    .ASCIZ  @EXPCTD  RECEVD@
     2 103146      102     125     123  EH110:  .ASCIZ  @BUSADR@
     3 103155      040     122     115  EH111:  .ASCIZ  @ RMCS2   RMCS1@
     4
     5 103174      122     105     103  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
     6 103223      105     130     120  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
     7 103251      105     130     120  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CRLF>
     8 103277      123     124     101          .ASCIZ  @STATUS  STATUS  INDEX@
     9
    10 103325      107     104     101  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
    11 103364      122     115     103  EH337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CRLF>
    12 103423      137     137     137          .ASCII  @                          @<CRLF>
    13 103462      105     130     120          .ASCIZ  @EXPCTD--RECEVD   --BIT---ADRESS@
    14
    15 103521      122     115     105  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CRLF>
    16 103561      137     137     137          .ASCII  @                  WORD     BIT@<CRLF>
    17 103617      105     130     120          .ASCIZ  @EXPCTD--RECEVD  NUMBER  POSITON@
    18 103657      105     130     120  EH353:  .ASCII  @EXPCTD  RECEVD@<CRLF>
    19 103676      040     122     115          .ASCIZ  @ RMLA    RMLA    RMOF @
    20
    21 103725      040     122     115  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
    22 103772      040     040     040          .ASCIZ  @   RMAS@
    23 104002      040     122     115  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
    24 104047      040     040     040          .ASCIZ  @    RMEC1   RMEC2@
    25 104071      040     122     115  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
    26 104127      040     122     115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
    27                                  .EVEN
```

ERROR MESSAGE STRINGS

```
      1 104166  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
      2 104174  001276  000000          ED110:  .WORD   $BASE,0
      3 104200  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0
      4
      5 104206  001364  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
      6 104216  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0
      7
      8 104226  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0
      9
     10 104240  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
     11 104252  001140  001142  001444  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0
     12
     13 104262  001336  001346  001350  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
     14 104300  001340  001342  001344  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
     15 104314  001404  000000          .WORD   RMEC2I,0
     16 104320  001344  001372  001370  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
     17 104332  001362  001376  001364  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

ERROR MESSAGE STRINGS

```
      1 104344     000                    EF110:  .BYTE   0
      2 104345     000     000            EF111:  .BYTE   0,0
      3 104347     000     000     000    EF114:  .BYTE   0,0,0
      4 104352     000     000     000    EF336:  .BYTE   0,0,0,0
      5 104356     000     000     000    EF337:  .BYTE   0,0,0,0,0
      6
      7 104363     000     000     000    STSF:   .BYTE   0,0,0,0,0,0,0
```

ERROR MESSAGE STRINGS

```
                                BUFFER:
                                BUFONE: .BLKW   258.
                                BUFTWO: .BLKW   258.

                104372          .=BUFFER

                                HELP:
                200             .ASCII  <CRLF>
                200             .ASCII  <CRLF>
                114     111     123     .ASCII  @LIST OF TESTS@<CRLF>
                055     055     055     .ASCII  @------------@<CRLF>
                124     061     011     .ASCII  @T1     CONTROLLER ACCESS TEST@<CRLF>
                124     062     011     .ASCII  @T2     LOGICAL ADDRESS PLUGS TEST@<CRLF>
                124     063     011     .ASCII  @T3     UNIBUS INITIALIZE TEST@<CRLF>
                124     064     011     .ASCII  @T4     CONTROLLER CLEAR TEST@<CRLF>
                124     065     011     .ASCII  @T5     ERROR CLEAR TEST@<CRLF>
                124     066     011     .ASCII  @T6     DRIVE STATUS TEST@<CRLF>
                124     067     011     .ASCII  @T7     PRIMARY/SECONDARY ERROR TEST@<CRLF>
                124     061     060     .ASCII  @T10    DIAGNOSTIC MODE TEST@<CRLF>
                124     061     061     .ASCII  @T11    PACK ACKNOWLEDGE TEST@<CRLF>
                124     061     062     .ASCII  @T12    RECALIBRATE TEST@<CRLF>
                124     061     063     .ASCII  @T13    ABORT RECALIBRATE TEST@<CRLF>
                124     061     064     .ASCII  @T14    IVC RECALIBRATE TEST@<CRLF>
                124     061     065     .ASCII  @T15    IAE RECALIBRATE TEST@<CRLF>
                124     061     066     .ASCII  @T16    RECALIBRATE AT OFFSET@<CRLF>
                124     061     067     .ASCII  @T17    DRIVE CLEAR TEST@<CRLF>
                124     062     060     .ASCII  @T20    NOP TEST@<CRLF>
                124     062     061     .ASCII  @T21    OFFSET TEST@<CRLF>
                124     062     062     .ASCII  @T22    GO/ATA TEST@<CRLF>
                124     062     063     .ASCII  @T23    WRITE ATA TEST@<CRLF>
                124     062     064     .ASCII  @T24    ERROR/ATA TEST@<CRLF>
                124     062     065     .ASCII  @T25    PROGRAM INTERRUPT TEST@<CRLF>
                124     062     066     .ASCII  @T26    INHIBIT INTERRUPT TEST@<CRLF>
                124     062     067     .ASCII  @T27    RETURN TO CENTERLINE TEST@<CRLF>
                124     063     060     .ASCII  @T30    READ IN PRESET TEST@<CRLF>
                124     063     061     .ASCII  @T31    RMDC CLEAR OFFSET TEST@<CRLF>
                124     063     062     .ASCII  @T32    ILLEGAL FUNCTION TEST@<CRLF>
                124     063     063     .ASCII  @T33    INVALID COMMAND TEST@<CRLF>
                124     063     064     .ASCII  @T34    INVALID ADDRESS ERROR TEST@<CRLF>
                124     063     065     .ASCII  @T35    WRITE LOCK ERROR TEST@<CRLF>
                124     063     066     .ASCII  @T36    ERROR ABORT TESTS@<CRLF>
                124     063     067     .ASCII  @T37    RMR TEST@<CRLF>
                124     064     060     .ASCII  @T40    PARITY ERROR TEST@<CRLF>
                124     064     061     .ASCII  @T41    ILLEGAL REGISTER TEST@<CRLF>
                124     064     062     .ASCII  @T42    SEEK FIRST CYLINDER@<CRLF>
                124     064     063     .ASCII  @T43    SEEK LAST CYLINDER@<CRLF>
                124     064     064     .ASCII  @T44    SEEK PRIME CYLINDERS@<CRLF>
                124     064     065     .ASCII  @T45    SEEK ZERO DIFFERENCE@<CRLF>
                124     064     066     .ASCII  @T46    SEEK MAXIMUM DIFFERENCE FORWARD@<CRLF>
                124     064     067     .ASCII  @T47    SEEK ADJACENT FORWARD@<CRLF>
                124     065     060     .ASCII  @T50    SEEK ADJACENT REVERSE@<CRLF>
                124     065     061     .ASCII  @T51    SEEK INVALID SECTOR@<CRLF>
                124     065     062     .ASCII  @T52    SEEK INVALID TRACK@<CRLF>
                124     065     063     .ASCII  @T53    SEEK INVALID CYLINDER@<CRLF>
                124     065     064     .ASCII  @T54    IVC SEEK TEST@<CRLF>
                124     065     065     .ASCII  @T55    ABORT SEEK TEST@<CRLF>
                124     065     066     .ASCII  @T56    SEEK AT OFFSET@<CRLF>
```

ERROR MESSAGE STRINGS

```
     58 106544      124      065      067  .ASCII  @T57    LOOK AHEAD TEST@<CRLF>
     59 106570      124      066      060  .ASCII  @T60    SEARCH ON CYLINDER@<CRLF>
     60 106617      124      066      061  .ASCII  @T61    SEARCH OFF CYLINDER@<CRLF>
     61 106647      124      066      062  .ASCII  @T62    SEARCH INVALID SECTOR@<CRLF>
     62 106701      124      066      063  .ASCII  @T63    SEARCH INVALID TRACK@<CRLF>
     63 106732      124      066      064  .ASCII  @T64    SEARCH INVALID CYLINDER@<CRLF>
     64 106766      124      066      065  .ASCII  @T65    IVC SEARCH TEST@<CRLF>
     65 107012      124      066      066  .ASCII  @T66    ABORT SEARCH TEST@<CRLF>
     66 107040      124      066      067  .ASCII  @T67    SEARCH AT OFFSET@<CRLF>
     67 107065      200               .ASCII  <CRLF>
     68 107066      117      120      105  .ASCII  @OPERATIONAL SWITCH SETTINGS@<CRLF>
     69 107122      055      055      055  .ASCII  @--------------------------@<CRLF>
     70 107156      123      127      111  .ASCII  @SWITCH                   USE@<CRLF>
     71 107173      055      055      055  .ASCII  @------          ----------------------@<CRLF>
     72 107230      040      040      061  .ASCII  @  15            HALT ON ERROR@<CRLF>
     73 107254      040      040      061  .ASCII  @  14            LOOP ON TEST@<CRLF>
     74 107277      040      040      061  .ASCII  @  13            INHIBIT ERROR TYPEOUTS@<CRLF>
     75 107334      040      040      061  .ASCII  @  12            @<CRLF>
     76 107343      040      040      061  .ASCII  @  11            INHIBIT ITERATIONS@<CRLF>
     77 107374      040      040      061  .ASCII  @  10            BELL ON ERROR@<CRLF>
     78 107420      040      040      040  .ASCII  @   9            LOOP ON ERROR@<CRLF>
     79 107444      040      040      040  .ASCII  @   8            LOOP ON TEST IN SWR<7:0>@<CRLF>
     80 107503      040      040      040  .ASCII  @   7            TN128@<CRLF>
     81 107517      040      040      040  .ASCII  @   6            TN64@<CRLF>
     82 107532      040      040      040  .ASCII  @   5            TN32@<CRLF>
     83 107545      040      040      040  .ASCII  @   4            TN16@<CRLF>
     84 107560      040      040      040  .ASCII  @   3            TN8@<CRLF>
     85 107572      040      040      040  .ASCII  @   2            TN4@<CRLF>
     86 107604      040      040      040  .ASCII  @   1            TN2@<CRLF>
     87 107616      040      040      040  .ASCIZ  @   0            TN1@<CRLF>
     88
     89          000200                .END    200
```

SYMBOL TABLE

```
ABASE = 176700          AUNIT = 000000          CH    = 002000          EDT114  074430          EMS102  077414
ACDW1 = 000000          AUSWR = 000000          CHGADR  001326          EDT2    074416          EMS103  077443
ACDW2 = 000000          AVECT1= 120254          CHRCNT  064201          EDT223  074432          EMS104  077452
ACKSTS  053464          AVECT2= 000000          CKSWR = 104410          EDT336  074442          EMS105  077501
ACPUOP= 000000          A16   = 000400          CLKADR  001512          EDT337  074452          EMS106  077530
ADDW0 = 000000          A17   = 001000          CLKVCT  001514          EDT344  074462          EMS11   075162
ADDW1 = 000000          BACK  = 000001          CLR   = 000040          EDT353  074474          EMS110  077547
ADDW10= 000000          BADTMO  005334          CLRSTS  052604          ED1     104166          EMS111  077576
ADDW11= 000000          BAI   = 000010          CMNSTA  007402          ED110   104174          EMS112  077607
ADDW12= 000000          BB00  = 000001          CMPERR  050562          ED111   104200          EMS113  077673
ADDW13= 000000          BB01  = 000002          CNSL01  066567          ED114   104206          EMS114  077723
ADDW14= 000000          BB02  = 000004          CNSL02  066577          ED223   104216          EMS115  077747
ADDW15= 000000          BB03  = 000010          CNSL03  066641          ED336   104226          EMS116  077752
ADDW2 = 000000          BB04  = 000020          CNSL04  066650          ED337   104240          EMS117  100002
ADDW3 = 000000          BB05  = 000040          CNSL07  066704          ED353   104252          EMS12   075173
ADDW4 = 000000          BB06  = 000100          CNSL08  067046          EECC  = 000020          EMS120  100024
ADDW5 = 000000          BB07  = 000200          CNSL09  067047          EFT1    074476          EMS121  100024
ADDW6 = 000000          BB08  = 000400          CNTCLR  052466          EFT110  074514          EMS122  100067
ADDW7 = 000000          BB09  = 001000          COMMA   066533          EFT111  074516          EMS123  100124
ADDW8 = 000000          BIT0  = 000001          CONT  = 000100          EFT114  074520          EMS124  100167
ADDW9 = 000000          BIT00 = 000001          CPSAVE  063352          EFT2    074506          EMS125  100221
ADEVCT= 000000          BIT01 = 000002          CR    = 000015          EFT223  074522          EMS126  100264
ADEVM = 000000          BIT02 = 000004          CRLF  = 000200          EFT336  074532          EMS127  100327
ADR   = 000001          BIT03 = 000010          CYLMSK= 001777          EFT337  074542          EMS13   075234
AENV  = 000000          BIT04 = 000020          DBCK  = 100000          EFT344  074552          EMS130  100366
AENVM = 000000          BIT05 = 000040          DBEN  = 040000          EFT353  074562          EMS131  100424
AFATAL= 000000          BIT06 = 000100          DBL   = 002000          EF110   104344          EMS132  100464
ALL     066522          BIT07 = 000200          DCK   = 100000          EF111   104345          EMS133  100510
AMADR1= 000000          BIT08 = 000400          DDISP = 177570          EF114   104347          EMS134  100533
AMADR2= 000000          BIT09 = 001000          DEBL  = 020000          EF336   104352          EMS135  100575
AMADR3= 000000          BIT1  = 000002          DEVSEL  050774          EF337   104356          EMS136  100650
AMADR4= 000000          BIT10 = 002000          DISPLA  001156          EHT1    074256          EMS137  100675
AMAMS1= 000000          BIT11 = 004000          DISPRE  000174          EHT110  074300          EMS14   075245
AMAMS2= 000000          BIT12 = 010000          DLT   = 100000          EHT111  074304          EMS140  100732
AMAMS3= 000000          BIT13 = 020000          DMD   = 000001          EHT112  074310          EMS141  100753
AMAMS4= 000000          BIT14 = 040000          DONE    067750          EHT114  074314          EMS142  101001
AMSGAD= 000000          BIT15 = 100000          DPE   = 000010          EHT2    074270          EMS143  101016
AMSGLG= 000000          BIT2  = 000004          DPEHI = 040000          EHT223  074320          EMS144  101044
AMSGTY= 000000          BIT3  = 000010          DPELO = 020000          EHT256  074332          EMS145  101054
AMTYP1= 000000          BIT4  = 000020          DPR   = 000400          EHT336  074344          EMS146  101103
AMTYP2= 000000          BIT5  = 000040          DRIVES  067244          EHT337  074356          EMS147  101122
AMTYP3= 000000          BIT6  = 000100          DRQ   = 004000          EHT344  074370          EMS15   075256
AMTYP4= 000000          BIT7  = 000200          DRVCLR= 000010          EHT353  074402          EMS150  101137
AOE   = 001000          BIT8  = 000400          DRVSTS  056022          EH1     103127          EMS151  101156
APASS = 000000          BIT9  = 001000          DRY   = 000200          EH110   103146          EMS152  101202
APE   = 100000          BLNKS1  070000          DSWR  = 177570          EH111   103155          EMS153  101226
APRIOR= 000000          BLNKS2  067777          DTE   = 010000          EH114   103174          EMS154  101250
APTCSU= 000040          BLNKS3  067776          DTO   = 010000          EH223   103223          EMS155  101264
APTENV= 000001          BLNKS4  067775          DVA   = 004000          EH256   103251          EMS156  101304
APTSIZ= 000200          BOTADR  064176          DVC   = 000200          EH336   103325          EMS157  101322
APTSPO= 000100          BOTFLG  064200          EBL   = 020000          EH337   103364          EMS16   075300
ARGS  = 000002          BPTVEC= 000014          ECH   = 000100          EH344   103521          EMS160  101337
ASWREG= 000000          BSE   = 100000          ECI   = 004000          EH353   103657          EMS161  101354
ATA   = 100000          BUFFER  104372          ECRC  = 001000          EMS1    074564          EMS162  101404
ATESTN= 000000          BUFONE  104372          EDT1    074406          EMS10   075120          EMS163  101404
ATNMSK= 000377          BUFTWO  105376          EDT110  074424          EMS100  077340          EMS164  101430
ATNTBL  070102          CC    = 004000          EDT111  074426          EMS101  077366          EMS165  101511
```

SYMBOL TABLE

EMS166	101525	EMS40	076067	EMT121	071424	EMT203	072310	EMT266	073142
EMS167	101552	EMS41	076116	EMT122	071434	EMT204	072322	EMT267	073154
EMS17	075330	EMS42	076144	EMT123	071444	EMT205	072340	EMT27	070336
EMS170	101577	EMS43	076173	EMT124	071454	EMT206	072350	EMT270	073172
EMS171	101626	EMS44	076222	EMT125	071464	EMT207	072360	EMT271	073206
EMS172	101661	EMS45	076275	EMT126	071474	EMT21	070256	EMT272	073224
EMS173	101701	EMS46	076340	EMT127	071506	EMT210	072372	EMT273	073242
EMS174	101716	EMS47	076367	EMT13	070200	EMT211	072400	EMT274	073260
EMS175	101724	EMS5	074756	EMT130	071520	EMT212	072414	EMT275	073276
EMS176	101752	EMS50	076416	EMT131	071532	EMT213	072430	EMT276	073310
EMS177	102001	EMS51	076445	EMT132	071544	EMT214	072440	EMT277	073324
EMS2	074633	EMS52	076473	EMT133	071556	EMT215	072460	EMT3	070124
EMS20	075370	EMS53	076505	EMT134	071570	EMT216	072472	EMT30	070346
EMS200	102023	EMS54	076517	EMT135	071602	EMT217	072502	EMT300	073342
EMS201	102040	EMS55	076541	EMT136	071614	EMT22	070266	EMT301	073362
EMS202	102106	EMS56	076570	EMT137	071626	EMT220	072512	EMT302	073404
EMS203	102136	EMS57	076645	EMT14	070212	EMT221	072522	EMT303	073416
EMS204	102151	EMS6	075006	EMT140	071636	EMT222	072532	EMT304	073430
EMS205	102177	EMS60	076673	EMT141	071646	EMT223	072542	EMT305	073442
EMS206	102211	EMS61	076706	EMT142	071656	EMT224	072552	EMT306	073454
EMS207	102236	EMS62	076735	EMT143	071666	EMT225	072566	EMT307	073464
EMS21	075413	EMS63	076753	EMT144	071676	EMT226	072606	EMT31	070356
EMS210	102263	EMS64	077001	EMT145	071706	EMT227	072616	EMT310	073504
EMS211	102274	EMS65	077017	EMT146	071716	EMT23	070276	EMT311	073516
EMS212	102312	EMS66	077065	EMT147	071726	EMT230	072632	EMT312	073530
EMS213	102335	EMS67	077135	EMT15	070220	EMT231	072636	EMT313	073542
EMS214	102360	EMS7	075053	EMT150	071736	EMT232	072646	EMT314	073562
EMS215	102376	EMS70	077162	EMT151	071746	EMT233	072650	EMT315	073574
EMS216	102417	EMS71	077174	EMT152	071760	EMT234	072652	EMT316	073606
EMS217	102447	EMS72	077207	EMT153	071772	EMT235	072654	EMT317	073620
EMS22	075436	EMS73	077217	EMT154	072010	EMT236	072656	EMT32	070366
EMS220	102502	EMS74	077234	EMT155	072026	EMT237	072660	EMT320	073630
EMS221	102522	EMS75	077253	EMT156	072040	EMT24	070306	EMT321	073640
EMS222	102572	EMS76	077261	EMT157	072052	EMT240	072662	EMT322	073650
EMS223	102623	EMS77	077312	EMT16	070226	EMT241	072664	EMT323	073656
EMS224	102700	EMTVEC=	000030	EMT160	072064	EMT242	072666	EMT324	073666
EMS225	102752	EMT1	070112	EMT161	072074	EMT243	072670	EMT325	073700
EMS226	102775	EMT10	070162	EMT162	072106	EMT244	072672	EMT326	073712
EMS227	103024	EMT100	071166	EMT163	072120	EMT245	072674	EMT327	073730
EMS23	075452	EMT101	071204	EMT164	072130	EMT246	072676	EMT33	070376
EMS230	103035	EMT102	071222	EMT165	072136	EMT247	072706	EMT330	073742
EMS231	103064	EMT103	071232	EMT166	072144	EMT25	070316	EMT331	073752
EMS232	103102	EMT104	071244	EMT167	072152	EMT250	072720	EMT332	073764
EMS24	075500	EMT105	071262	EMT17	070236	EMT251	072734	EMT333	073776
EMS25	075527	EMT106	071272	EMT170	072160	EMT252	072742	EMT334	074014
EMS26	075544	EMT107	071302	EMT171	072172	EMT253	072750	EMT335	074032
EMS27	075565	EMT11	070166	EMT172	072202	EMT254	072766	EMT336	074052
EMS3	074650	EMT110	071322	EMT173	072212	EMT255	072776	EMT337	074062
EMS30	075576	EMT111	071334	EMT174	072222	EMT256	073006	EMT34	070406
EMS31	075606	EMT112	071342	EMT175	072234	EMT257	073016	EMT340	074072
EMS32	075635	EMT113	071350	EMT176	072242	EMT26	070326	EMT341	074104
EMS33	075664	EMT114	071364	EMT177	072250	EMT260	073030	EMT342	074112
EMS34	075712	EMT115	071364	EMT2	070116	EMT261	073042	EMT343	074124
EMS35	075763	EMT116	071374	EMT20	070246	EMT262	073062	EMT344	074136
EMS36	076012	EMT117	071404	EMT200	072252	EMT263	073072	EMT345	074150
EMS37	076041	EMT12	070172	EMT201	072264	EMT264	073102	EMT346	074160
EMS4	074713	EMT120	071414	EMT202	072276	EMT265	073122	EMT347	074174

SYMBOL TABLE

```
EMT35   070416          FER    = 000020        IOTVEC= 000020        PAKACK= 000022        RMBAI   001342
EMT350  074206          FIND   = 000001        IPCK0  = 000001        PAR    = 000010        RMBAO   001416
EMT351  074224          FMT16  = 010000        IPCK1  = 000002        PAT    = 000020        RMCS1  = 000000
EMT352  074234          FNCDTB  070002        IPCK2  = 000004        PDA    = 000400        RMCS1I  001336
EMT353  074240          FNCMSK= 000077        IPCK3  = 000010        PFECH   064244         RMCS10  001412
EMT354  074250          F0     = 000002        IR     = 000100        PFECH1  064254         RMCS2  = 000010
EMT36   070426          F1     = 000004        IVC    = 010000        PFECH2  064342         RMCS2I  001346
EMT37   070436          F2     = 000010        LBC    = 002000        PFECH3  064360         RMCS20  001422
EMT4    070132          F3     = 000020        LBT    = 002000        PFECH4  064366         RMCS3  = 000052
EMT40   070446          F4     = 000040        LF     = 000012        PGE    = 002000        RMCS3I  001410
EMT41   070454          GET      042546        LODEV   067152         PGM    = 001000        RMCS30  001464
EMT42   070464          GETBUF   001336        LS     = 000004        PHA    = 000200        RMDA   = 000006
EMT43   070476          GETINX   001516        LSC    = 004000        PIP    = 020000        RMDAI   001344
EMT44   070506          GETSTS   042462        LST    = 000002        PIRQ   = 177772        RMDAO   001420
EMT45   070516          GO     = 000001        LSTRK   001334         PIRQVE= 000240        RMDB   = 000022
EMT46   070526          GTSWR  = 104407        MANUAL  001330         PLFS   = 002000        RMDBI   001360
EMT47   070536          HCE    = 000200        MCLK   = 004000        PRIERR  043564         RMDBO   001434
EMT5    070140          HCI    = 002000        MCPE   = 020000        PR0    = 000000        RMDC   = 000034
EMT50   070544          HCRC   = 000400        MDF    = 000100        PR1    = 000040        RMDCI   001372
EMT51   070554          HELP     104372        MDPE   = 000400        PR2    = 000100        RMDCO   001446
EMT52   070566          HT     = 000011        MI     = 000004        PR3    = 000140        RMDS   = 000012
EMT53   070604          IAE    = 002000        MINSTR  067443         PR4    = 000200        RMDSI   001350
EMT54   070622          IBSAVE   063354        MOC    = 000400        PR5    = 000240        RMDSO   001424
EMT55   070632          IDXMSK= 000077        MOH    = 020000        PR6    = 000300        RMDT   = 000026
EMT56   070644          IE     = 000100        MOL    = 010000        PR7    = 000340        RMDTI   001364
EMT57   070662          ILF    = 000001        MRD    = 002000        PS     = 177776        RMDTO   001440
EMT6    070146          ILF02  = 000002        MS     = 000040        PSEL   = 002000        RMEC1  = 000044
EMT60   070672          ILF24  = 000024        MSC    = 000002        PSW    = 177776        RMEC1I  001402
EMT61   070710          ILF26  = 000026        MSDRVS  067070         PULL    067655         RMEC10  001456
EMT62   070730          ILF30  = 000030        MSE    = 100000        PUT     043016         RMEC2  = 000046
EMT63   070744          ILF32  = 000032        MSER   = 000200        PUTBUF  001412         RMEC2I  001404
EMT64   070754          ILF34  = 000034        MSGDRV  067104         PUTINX  001545         RMEC20  001460
EMT65   070774          ILF36  = 000036        MSHELP  066536         PWRVEC= 000024         RMER1  = 000014
EMT66   071014          ILF40  = 000040        MSMNUL  067300         QUES    066527         RMER1I  001352
EMT67   071026          ILF42  = 000042        MSUNIT  067360         RCLSTS  054260         RMER10  001426
EMT7    070154          ILF44  = 000044        MUR    = 001000        RD     = 000070        RMER2  = 000042
EMT70   071040          ILF46  = 000046        MWD    = 000010        RDCHR  = 104411         RMER2I  001400
EMT71   071050          ILF54  = 000054        MWP    = 000010        RDLIN  = 104412         RMER20  001454
EMT72   071060          ILF56  = 000056        MXF    = 001000        RDOCT  = 104413         RMHR   = 000036
EMT73   071076          ILF64  = 000064        N       067771         RDY    = 000200         RMHRI   001374
EMT74   071114          ILF66  = 000066        NDTMSK= 115760         READY   007574         RMHRO   001450
EMT75   071124          ILF74  = 000074        NED    = 010000        RECAL  = 000006         RMLA   = 000020
EMT76   071144          ILF76  = 000076        NEM    = 004000        RESREG= 104415         RMLAI   001356
EMT77   071154          ILR    = 000002        NONE    067273         RESVEC= 000010         RMLAO   001432
EQUALS  066520          ILRG50= 000050        NOP    = 000000        RET     067720         RMMR1  = 000024
ERR    = 040000         ILRG52= 000052        NOTAVL  067204         REX    = 010000        RMMR1I  001362
ERRMSB  064174          ILRG54= 000054        NOTPRS  067167         RG     = 040000        RMMR10  001436
ERROR  = 104000         ILRG56= 000056        NOTRM   067135         RH     = 000072        RMMR2  = 000040
ERRTYP  063356          ILRG60= 000060        NSA    = 100000        RIP    = 000020        RMMR2I  001376
ERRVEC= 000004          ILRG62= 000062        OCC    = 100000        RLEASE= 000012         RMMR20  001452
ERTY00  064202          ILRG64= 000064        OFD    = 000200        RMAS   = 000016         RMOF   = 000032
ERTY01  064207          ILRG66= 000066        OFFSET= 000014         RMASI   001354         RMOFI   001370
ERTY02  064217          ILRG70= 000070        OM     = 000001        RMASO   001430         RMOFO   001444
ERTY03  064226          ILRG72= 000072        OPE    = 020000        RMBA   = 000004         RMR    = 000004
ERTY04  064234          ILRG74= 000074        OPI    = 020000        RMBAE  = 000050         RMSN   = 000030
ERTY05  064237          ILRG76= 000076        OR     = 000200        RMBAEI  001406         RMSNI   001366
ESRC   = 004000         INSERT  067617        PACACK= 000022        RMBAEO  001462         RMSNO   001442
```

SYMBOL TABLE

```
RMWC  = 000002    SW02  = 000004    TST24  017772    WC    = 000040    $EOSP  041256
RMWCI   001340    SW03  = 000010    TST25  020364    WCD   = 000050    $ERFLG 001117
RMWCO   001414    SW04  = 000020    TST26  021054    WCE   = 040000    $ERMAX 001131
RQA   = 100000    SW05  = 000040    TST27  021422    WCEHI = 010000    $ERROR 062762
RQB   = 040000    SW06  = 000100    TST3   010532    WCELO = 004000    $ERRPC 001132
RTC   = 000016    SW07  = 000200    TST30  021744    WCF   = 000040    $ERRTB 001574
R6    =%000006    SW08  = 000400    TST31  022326    WCH   = 000052    $ERTTL 001126
R7    =%000007    SW09  = 001000    TST32  022644    WD    = 000060    $ESCAP 001210
SADMSK= 000377    SW1   = 000002    TST33  023430    WH    = 000062    $ETABL 001242
SAVREG= 104414    SW10  = 002000    TST34  024154    WLE   = 004000    $ETEND 001326
SA1   = 000001    SW11  = 004000    TST35  024744    WRL   = 004000    $FATAL 001224
SA16  = 000020    SW12  = 010000    TST36  025530    XSIZ    006356    $FFLG  066516
SA2   = 000002    SW13  = 020000    TST37  026062    XXDP    001332    $FILLC 001172
SA4   = 000004    SW14  = 040000    TST4   011402    Y       067773    $FILLS 001171
SA8   = 000010    SW15  = 100000    TST40  026522    $APTHD  001100    $GDADR 001134
SC    = 100000    SW2   = 000004    TST41  026742    $ATYC   066276    $GDDAT 001140
SCHSTS  056624    SW3   = 000010    TST42  027300    $ATY1   066252    $GET42 041502
SCOPE = 000004    SW4   = 000020    TST43  027572    $ATY3   066260    $GTSWR 064772
SCTMSK= 003700    SW5   = 000040    TST44  030064    $ATY4   066270    $GT42P 041476
SC0   = 000100    SW6   = 000100    TST45  030376    $AUTOB  001150    $HD   = 000000
SC1   = 000200    SW7   = 000200    TST46  030704    $BASE   001276    $HIBTS 001100
SC2   = 000400    SW8   = 000400    TST47  031172    $BDADR  001136    $HIOCT 065764
SC3   = 001000    SW9   = 001000    TST5   011740    $BDDAT  001142    $ICNT  001120
SC4   = 002000    SYSTAT  067112    TST50  031474    $BELL   001212    $ILLUP 066234
SEARCH= 000030    TADMSK= 177400    TST51  032000    $BIN    061062    $INTAG 001151
SECERR  044416    TAG   = 020000    TST52  032416    $CDW1   001302    $ITEMB 001130
SEEK  = 000004    TAGADR= 001114    TST53  033014    $CDW2   001304    $LF    001220
SEKSTS  051206    TAP   = 040000    TST54  033406    $CHARC  062066    $LFLG  066515
SHUT    065642    TA1   = 000400    TST55  033746    $CKSWR  064702    $LPADR 001122
SIZCLK  043246    TA2   = 001000    TST56  034310    $CMTAG  001114    $LPERR 001124
SKI   = 040000    TA4   = 002000    TST57  034634    $CM3  = 000000    $MADR1 001254
SSE   = 000040    TA8   = 004000    TST6   012146    $CM4  = 000005    $MADR2 001260
SSEI  = 001000    TBITVE= 000014    TST60  035354    $CNTLC  065600    $MADR3 001264
SSF   = 020000    TIMOUT  043370    TST61  035776    $CNTLG  065612    $MADR4 001270
STACK = 001100    TKVEC = 000060    TST62  036440    $CNTLU  065605    $MAIL  001222
STANDA  006646    TPVEC = 000064    TST63  037040    $CPUOP  001250    $MAMS1 001252
START   005426    TRAPVE= 000034    TST64  037420    $CRLF   001217    $MAMS2 001256
START1  005416    TRE   = 040000    TST65  037774    $DBLK   061300    $MAMS3 001262
START2  005432    TRTVEC= 000014    TST66  040354    $DDW0   001306    $MAMS4 001266
STCDRV  060210    TST   = 010000    TST67  040720    $DDW1   001310    $MBADR 001102
STIMER  043532    TSTNMB  064172    TST7   012462    $DDW2   001312    $MFLG  066514
STKLMT= 177774    TSTPRP  041532    TYPBN = 104406    $DDW3   001314    $MNEW  065630
STOP    062734    TSTQUE  001466    TYPDS = 104405    $DDW4   001316    $MSGAD 001236
STSD1   104262    TST1    007736    TYPE  = 104401    $DDW5   001320    $MSGLG 001240
STSD2   104300    TST10   012642    TYPOC = 104402    $DDW6   001322    $MSGTY 001222
STSD3   104320    TST11   013714    TYPON = 104404    $DDW7   001324    $MSWR  065617
STSD4   104332    TST12   014214    TYPOS = 104403    $DEVCT  001232    $MTYP1 001253
STSF    104363    TST13   014424    UNS   = 040000    $DEVM   001300    $MTYP2 001257
STSH1   103725    TST14   014732    UNTMSK= 000007    $DOAGN  041522    $MTYP3 001263
STSH2   104002    TST15   015242    UNTOFF  067223    $DTBL   061270    $MTYP4 001267
STSH3   104071    TST16   015504    UNTON   067234    $ENDAD  041512    $MXCNT 062554
STSH4   104127    TST17   015762    UPE   = 020000    $ENDCT  041350    $NULL  001170
SWR     001154    TST2    010136    USE   = 040000    $ENULL  041526    $NWTST= 000001
SWREG   000176    TST20   016232    U0    = 000001    $ENV    001242    $OCNT  061532
SW0   = 000001    TST21   016462    U1    = 000002    $ENVM   001243    $OMODE 061534
SW00  = 000001    TST22   017006    U2    = 000004    $EOP    041314    $OVER  062540
SW01  = 000002    TST23   017426    VV    = 000100    $EOPCT  041342    $PASS  001230
```

```
CZRNDA0 RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE 58-6                SEQ 0286
SYMBOL TABLE

$PASTM  001106          $SCOPE  062072          $TKQEN= 064401          $TPS    001164          $TYPON  061350
$POWER  066242          $SETUP= 000137          $TKQIN  064374          $TRAP   065766          $TYPOS  061310
$PWRDN  066074          $STUP = 177777          $TKQOU  064376          $TRAP2  066026          $UNIT   001234
$PWRMG  066230          $SVLAD  062504          $TKQSR  064400          $TRP  = 000016          $UNITM  001110
$PWRUP  066146          $SVPC = 000210          $TKS    001160          $TRPAD  066040          $USWR   001246
$QUES   001216          $SWR  = 167400          $TKSRV  064452          $TSTM   001104          $VECT1  001272
$RDCHR  065244          $SWREG  001244          $TMP0   001174          $TSTNM  001116          $VECT2  001274
$RDLIN  065334          $SWRMK= 000000          $TMP1   001176          $TTYIN  065570          $XOFF = 000023
$RDOCT  065664          $SW08T  062556          $TMP2   001200          $TYPBN  061010          $XON  = 000021
$RDSZ = 000010          $TESTN  001226          $TMP3   001202          $TYPDS  061064          $XTSTR  062114
$RESRE  060752          $TIMES  001206          $TMP4   001204          $TYPE   061536          $$GET4= 000000
$RM80   067130          $TKB    001162          $TN   = 000070          $TYPEC  061750          $$SW08= 000070
$RTNAD  041524          $TKCNT  064372          $TPB    001166          $TYPEX  062070          $OFILL  061533
$SAVRE  060714          $TKINT  064402          $TPFLG  001173          $TYPOC  061334          .$X   = 001100
$SAVR6  066240

. ABS.  107631     000
        000000     001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  61952 WORDS  ( 242 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
CZRNDA.BIC,CZRNDA/C=CZRNDA.DOC,CZRNDA,[20,0]SYSMAC/M
```

```
$$GET4  14-20   14-20#
$$SW08  39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1
        39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1    39-1#   39-1#
        39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#
        39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#
        39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#
        39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#   39-1#
$OFILL  37-1    37-1#   37-1*   37-1*
$40CAT  39-1    40-1
$APTHD   5-8     5-8#
$ASTAT  46-1    46-1
$ATY1   46-1#
$ATY3   38-1    46-1#
$ATY4   40-1    46-1#
$ATYC   46-1    46-1#
$AUTOB   6-0#   10-29*  42-1    42-1    42-1
$BASE    6-0#   10-75   11-33   11-42*  12-52   13-2    13-33   13-105  13-210  13-252  13-277  13-328  13-346  13-448
        13-473  13-490  13-514  13-536  13-561  13-583  13-613  13-631  13-662  13-698  13-733  13-765  13-830  13-884
        13-915  13-959  13-988  13-:54  13-;21  13-;87  13-<53  13-<95  13-=27  13-=60  13->11  13->41  13->71  13-?05
        13-?39  13-?67  13-@01  13-@35  13-@84  13-A28  13-A70  13-B04  13-B41  13-B77  13-C70  13-D14  13-D65  13-E10
        13-E51  13-E91  13-F30  13-F69  17-36   18-30   20-12   25-23   27-15   41-25   56-2
$BDADR   6-0#   56-8
$BDDAT   6-0#   13-141* 13-142* 13-144  13-149* 13-153* 13-158  13-163* 13-164* 13-168* 13-174* 13-178* 13-179* 13-181
        13-186* 13-190* 13-191* 13-193  13-198* 13-199* 13-202* 13-203* 13-205  13-264* 13-266* 13-270* 13-271* 13-290*
        13-296* 13-302* 13-309* 13-318* 13-323* 13-357* 13-362* 13-363* 13-368* 13-369* 13-374* 13-375* 13-380* 13-381*
        13-386* 13-387* 13-397* 13-402* 13-403* 13-408* 13-409* 13-414* 13-415* 13-420* 13-421* 13-426* 13-427* 13-432*
        13-433* 13-442* 13-444* 13-644* 13-645* 13-650* 13-651* 13-673* 13-674* 13-686* 13-687* 13-708* 13-709* 13-725*
        13-726* 13-743* 13-744* 13-756* 13-757* 13-898* 13-899* 13-903* 13-904* 13-936* 13-937* 13-943* 13-948* 13-971*
        13-972* 13-982* 13-983* 13-:31* 13-:32* 13-:33  13-:42* 13-:43* 13-:44  13-:98* 13-:99* 13-;00  13-;09* 13-;10*
        13-;11  13-;64* 13-;65* 13-;66  13-;75* 13-;76* 13-;77  13-<30* 13-<31* 13-<32  13-<41* 13-<42* 13-<43  13-=23*
        13-=52* 13-=82* 13-=83* 13-=84  13-A95* 13-A96* 13-B24* 13-B25* 13-C49* 13-F21* 13-F22* 13-F52* 13-F53* 21-47*
        21-48*  21-51   21-69*  21-75*  21-91*  21-108* 21-124* 21-152* 22-40*  22-41*  22-53*  22-54*  22-64*  22-65*
        22-67   22-85*  22-86*  22-87   22-107* 22-108* 22-109  22-140* 22-141* 22-142  22-156* 22-157* 22-158  22-170*
        22-171* 22-183* 22-184* 22-200* 22-201* 22-226* 22-227* 22-241* 22-242* 22-256* 22-257* 22-271* 22-272* 22-284*
        22-285* 22-314* 22-315* 22-329* 22-330* 22-344* 22-345* 22-361* 22-362* 22-387* 22-388* 22-402* 22-403* 22-417*
        22-418* 22-432* 22-433* 22-460* 22-461* 22-483* 22-484* 22-498* 22-499* 22-520* 22-521* 23-10*  23-34*  23-95*
        23-96*  23-102  23-110  23-126* 23-127* 23-128  23-154* 23-155* 23-156  23-171* 23-172  23-184* 23-185  24-39*
        24-54*  26-37*  26-71*  26-72*  26-73   26-93*  26-104* 26-105* 26-111  26-122  26-135* 26-148* 26-173* 26-187*
        26-188* 26-201* 26-217* 28-29*  28-30*  28-32   28-42*  28-51*  28-59   28-69*  28-70*  28-79*  28-80*  28-82
        28-92*  28-101* 28-102* 28-104  28-114* 28-115* 28-124* 28-125* 28-127  29-27*  29-41*  29-57*  29-71*  29-85*
        29-99*  29-113* 30-35*  30-49*  30-65*  30-82*  30-103* 30-120* 30-134* 30-159* 30-174* 30-189* 30-205* 30-222*
        30-240* 30-254* 30-270* 31-16*  31-17*  31-19   31-28*  31-29*  31-31   31-41*  31-50*  31-57*  31-60   31-69*
        31-70*  31-72   31-81*  31-82*  31-84   31-94*  31-103* 32-58*  32-76*  32-112* 32-113* 32-114  32-134* 32-135*
        32-138  32-158* 32-175* 32-191* 32-214* 32-232* 32-248* 32-262* 33-46*  33-62*  33-78*  33-98*  33-112* 56-1
        56-6    56-8    56-10   56-11
$BELL    6-0#   40-1    40-1    40-1    42-1    42-1    42-1
$BIN    35-1    35-1#   35-1*   35-1*
$CDW1    6-0#
$CDW2    6-0#
$CHARC  38-1    38-1#   38-1*   38-1*   38-1*
```

CROSS REFERENCE TABLE (CREF V04.00)

```
$CKSWR  42-1#   44-1    44-1
$CM3     6-0     6-0#
$CM4     6-0     6-0     6-0     6-0     6-0     6-0     6-0     6-0     6-0     6-0     6-0#    6-0#    6-0#    6-0#
        6-0#    6-0#
$CMTAG   6-0#   10-24   10-24   10-24   10-24   10-24   10-24   10-24
$CNTLC  42-1    42-1    42-1    42-1    42-1#
$CNTLG  42-1    42-1#
$CNTLU  42-1    42-1    42-1#
$CPUOP   6-0#
$CRLF    6-0#   10-49   10-80   10-129  11-20   11-28   11-79   11-117  12-25   12-58   13-46   13-52   13-97   14-20
        38-1    38-1    38-1    40-1    40-1    40-1    41-20   41-53   41-96   41-106  41-118  41-126  41-146  42-1
        42-1    42-1    42-1
$DBLK   36-1    36-1    36-1#
$DDW0    6-0#
$DDW1    6-0#
$DDW2    6-0#
$DDW3    6-0#
$DDW4    6-0#
$DDW5    6-0#
$DDW6    6-0#
$DDW7    6-0#
$DEVCT   6-0#
$DEVM    6-0#   10-67*  10-78   10-106*  10-110*  11-80*  11-87*  11-114*  11-115  12-4    12-38   13-29*  42-6*
$DOAGN  14-20   14-20   14-20#
$DTBL   36-1    36-1#
$ENDAD   5-5    10-29   14-20#  40-1
$ENDCT  10-24   14-20#
$ENULL  14-20#
$ENV     6-0#   10-29   38-1    40-1    46-1    46-1
$ENVM    6-0#   10-24   10-71   38-1    38-1    46-1
$EOP    14-20#  42-7
$EOPCT  10-24*  14-20   14-20#
$EOSP   13-312  14-9#
$ERFLG   6-0#   39-1    39-1    39-1    39-1    39-1    39-1*   40-1    40-1    40-1*
$ERMAX   6-0#   10-24*  39-1    39-1    39-1    39-1*
$ERROR  10-24   40-1#
$ERRPC   6-0#   40-1    40-1    40-1    40-1*   40-1*   41-46
$ERRTB   8-0#   41-64
$ERTTL   6-0#   14-20   14-20   14-20*  40-1    40-1    40-1*
$ESCAP   6-0#   10-24*  39-1*   40-1    40-1    40-1    40-1
$ETABL   6-0#
$ETEND   5-8     6-0#
$FATAL   6-0#   46-1*
$FFLG   46-1    46-1#   46-1*   46-1*   46-1*
$FILLC   6-0#   38-1    38-1    38-1
$FILLS   6-0#   38-1    38-1
$GDADR   6-0#   56-8
$GDDAT   6-0#   13-143* 13-144  13-148*  13-157*  13-158  13-162*  13-180*  13-181  13-185*  13-192*  13-193  13-197*  13-204*
        13-205  13-265* 13-287* 13-288*  13-289*  13-297* 13-303*  13-310*  13-316* 13-317*  13-324*  13-356* 13-364*  13-370*
        13-376* 13-382* 13-388* 13-396*  13-404*  13-410* 13-416*  13-422*  13-428* 13-434*  13-443*  13-646* 13-652*  13-672*
        13-688* 13-710* 13-727* 13-745*  13-755*  13-897* 13-905*  13-939*  13-944* 13-949*  13-973*  13-984* 13-:29*  13-:30*
        13-:33  13-:37* 13-:40* 13-:41*  13-:44   13-:96* 13-:97*  13-;00   13-;04* 13-;07*  13-;08*  13-;11  13-;62*  13-;63*
        13-;66  13-;70* 13-;73* 13-;74*  13-;77   13-<28* 13-<29*  13-<32   13-<36* 13-<39*  13-<40*  13-<43  13-=22*  13-=51*
        13-=81* 13-=84  13-A94* 13-B26*  13-C48*  13-F20* 13-F51*  21-49*   21-50*  21-51   21-67*   21-68*  21-74*   21-76*
        21-88*  21-89*  21-90*  21-106*  21-107*  21-122* 21-123*  21-150*  21-151* 22-42*   22-55*   22-66*  22-67    22-79*
        22-84*  22-87   22-98*  22-106*  22-109   22-135* 22-138*  22-139*  22-142  22-154*  22-155*  22-158  22-165*  23-16*
```

CROSS REFERENCE TABLE (CREF V04.00)

	23-23*	23-24*	23-25*	23-26*	23-30*	23-32	23-94*	23-109*	23-110	23-120*	23-125*	23-128	23-138*	23-151*
	23-156	23-168*	23-169*	23-170*	23-172	23-182*	23-183*	23-185	24-40*	24-55*	26-36*	26-48*	26-68*	26-73
	26-92*	26-106*	26-107*	26-134*	26-147*	26-171*	26-172*	26-186*	26-199*	26-200*	26-215*	26-216*	28-31*	28-32
	28-41*	28-57*	28-59	28-68*	28-81*	28-82	28-91*	28-103*	28-104	28-113*	28-126*	28-127	29-25*	29-26*
	29-39*	29-40*	29-58*	29-59*	29-72*	29-73*	29-86*	29-87*	29-100*	29-101*	29-114*	29-115*	30-33*	30-34*
	30-47*	30-48*	30-63*	30-64*	30-80*	30-81*	30-101*	30-102*	30-118*	30-119*	30-132*	30-133*	30-157*	30-158*
	30-173*	30-175*	30-187*	30-188*	30-203*	30-204*	30-220*	30-221*	30-238*	30-239*	30-252*	30-253*	30-268*	30-269*
	31-18*	31-19	31-30*	31-31	31-40*	31-71*	31-72	31-83*	31-84	31-92*	32-56*	32-57*	32-74*	32-75*
	32-89*	32-109*	32-114	32-136*	32-137*	32-138	32-142	32-156*	32-157*	32-173*	32-174*	32-189*	32-190*	32-212*
	32-213*	32-230*	32-231*	32-246*	32-247*	32-260*	32-261*	33-44*	33-45*	33-60*	33-76*	33-77*	33-96*	33-97*
	33-110*	33-111*	56-1	56-6	56-8	56-10	56-11							
$GET42	12-34	12-43	13-30	14-20#										
$GT42P	14-20	14-20#												
$GTSWR	42-1#	44-1	44-1											
$HD	4-475	4-475	4-475											
$HIBTS	5-8#													
$HIOCT	43-1#	43-1*												
$ICNT	6-0#	39-1	39-1	39-1	39-1*	39-1*								
$ILLUP	45-1	45-1	45-1#											
$INTAG	6-0#	42-1	42-1	42-1	42-1	42-1*								
$ITEMB	6-0#	40-1	40-1	40-1	40-1	40-1	40-1*	40-1*	41-41					
$LF	6-0#	38-1	38-1	40-1	40-1	42-1	42-1	42-1						
$LFLG	46-1#	46-1*												
$LPADR	6-0#	10-24*	39-1	39-1	39-1	39-1*	39-1*	39-1*						
$LPERR	6-0#	10-24*	13-38*	39-1	39-1	39-1	39-1*	40-1						
$MADR1	6-0#													
$MADR2	6-0#													
$MADR3	6-0#													
$MADR4	6-0#													
$MAIL	5-8	5-8	6-0#	10-24	10-29	13-2	13-33	13-105	13-210	13-252	13-277	13-328	13-346	13-448
	13-473	13-490	13-514	13-536	13-561	13-583	13-613	13-631	13-662	13-698	13-733	13-765	13-830	13-884
	13-915	13-959	13-988	13-:54	13-;21	13-;87	13-<53	13-<95	13-=27	13-=60	13->11	13->41	13->71	13-?05
	13-?39	13-?67	13-@01	13-@35	13-@84	13-A28	13-A70	13-B04	13-B41	13-B77	13-C70	13-D14	13-D65	13-E10
	13-E51	13-E91	13-F30	13-F69	38-1	39-1	40-1							
$MAMS1	6-0#													
$MAMS2	6-0#													
$MAMS3	6-0#													
$MAMS4	6-0#													
$MBADR	5-8#													
$MFLG	46-1	46-1#	46-1*	46-1*										
$MNEW	42-1	42-1#												
$MSGAD	6-0#	46-1	46-1*											
$MSGLG	6-0#	46-1*												
$MSGTY	6-0#	46-1	46-1	46-1*	46-1*									
$MSWR	42-1	42-1#												
$MTYP1	6-0#													
$MTYP2	6-0#													
$MTYP3	6-0#													
$MTYP4	6-0#													
$MXCNT	39-1	39-1	39-1	39-1#										
$NULL	6-0#	38-1	38-1	38-1										
$NWTST	13-2	13-2#	13-2#	13-33	13-33#	13-33#	13-105	13-105#	13-105#	13-210	13-210#	13-210#	13-252	13-252#
	13-252#	13-277	13-277#	13-277#	13-328	13-328#	13-328#	13-346	13-346#	13-346#	13-448	13-448#	13-448#	13-473
	13-473#	13-473#	13-490	13-490#	13-490#	13-514	13-514#	13-514#	13-536	13-536#	13-536#	13-561	13-561#	13-561#
	13-583	13-583#	13-583#	13-613	13-613#	13-613#	13-631	13-631#	13-631#	13-662	13-662#	13-662#	13-698	13-698#
	13-698#	13-733	13-733#	13-733#	13-765	13-765#	13-765#	13-830	13-830#	13-830#	13-884	13-884#	13-884#	13-915

CZRNDAG RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE S-4 SEQ 0290

CROSS REFERENCE TABLE (CREF V04.00)

```
                13-915#  13-915#  13-959   13-959#  13-959#  13-988   13-988#  13-988#  13-:54   13-:54#  13-:54#  13-;21   13-;21#  13-;21#
                13-:87   13-:87#  13-:87#  13-<53   13-<53#  13-<53#  13-<95   13-<95#  13-<95#  13-=27   13-=27#  13-=27#  13-=60   13-=60#
                13-=60#  13->11   13->11#  13->11#  13->41   13->41#  13->41#  13->71   13->71#  13->71#  13-?05   13-?05#  13-?05#  13-?39
                13-?39#  13-?39#  13-?67   13-?67#  13-?67#  13-@01   13-@01#  13-@01#  13-@35   13-@35#  13-@35#  13-@84   13-@84#  13-@84#
                13-A28   13-A28#  13-A28#  13-A70   13-A70#  13-A70#  13-B04   13-B04#  13-B04#  13-B41   13-B41#  13-B41#  13-B77   13-B77#
                13-B77#  13-C70   13-C70#  13-C70#  13-D14   13-D14#  13-D14#  13-D65   13-D65#  13-D65#  13-E10   13-E10#  13-E10#  13-E51
                13-E51#  13-E51#  13-E91   13-E91#  13-E91#  13-F30   13-F30#  13-F30#  13-F69   13-F69#  13-F69#
$OCNT           37-1#    37-1*    37-1*
$OMODE          37-1     37-1#    37-1*    37-1*    37-1*    37-1*
$OVER           39-1     39-1     39-1     39-1     39-1#
$PASS            6-0#    10-24*   14-20    14-20    14-20    14-20*   14-20*   39-1     39-1     39-1
$PASTM           5-8#
$POWER          45-1     45-1#
$PWRDN          10-24    45-1     45-1#
$PWRMG          45-1#
$PWRUP          45-1     45-1#
$QUES            6-0#    38-1     38-1     40-1     40-1     42-1     42-1     42-1     42-1
$R2A            44-1
$RDCHR          42-1#    44-1     44-1
$RDDEC          44-1
$RDLIN          42-1#    44-1     44-1
$RDOCT          43-1#    44-1     44-1
$RDSZ           42-1     42-1#
$RESRE          34-1#    44-1
$RM80           10-91    41-28    47-19#
$RTNAD          14-20#
$SAVR6          45-1     45-1#    45-1*    45-1*    45-1*
$SAVRE          34-1#    44-1     44-1
$SCOPE          10-24    39-1#
$SETUP           4-769    4-769    4-769    4-769    4-769    4-769    4-769#   4-769#   4-769#   4-769#   4-769#   4-769#   4-769#   10-24
                10-24    10-24    10-24    10-24    10-24    10-24    10-24    10-24    10-24    10-24    10-24    10-29    10-29    10-29
                14-20    14-20    39-1     40-1     40-1     40-1     40-1     42-1     42-1     42-1     42-1     42-1
$STUP            4-769    4-769    4-769    4-769    4-769    4-769    4-769#   4-769#   4-769#   4-769#   4-769#   4-769#   4-769#   4-769#
                 4-769#   4-769#   4-769#   4-769#
$SVLAD          39-1     39-1#
$SVPC            5-5      5-5#
$SW08T          39-1     39-1#
$SWR             4-464#   4-475    4-476    4-476    4-476    4-476    4-476    4-476    4-476    4-476    6-0      6-0      6-0      10-24
                10-24    10-24    10-24    10-24    13-2     13-33    13-105   13-210   13-252   13-277   13-328   13-346   13-448   13-473
                13-490   13-514   13-536   13-561   13-583   13-613   13-631   13-662   13-698   13-733   13-765   13-830   13-884   13-915
                13-959   13-988   13-:54   13-;21   13-;87   13-<53   13-<95   13-=27   13-=60   13->11   13->41   13->71   13-?05   13-?39
                13-?67   13-@01   13-@35   13-@84   13-A28   13-A70   13-B04   13-B41   13-B77   13-C70   13-D14   13-D65   13-E10   13-E51
                13-E91   13-F30   13-F69   14-20    14-20    14-20    14-20    14-20    39-1     39-1     39-1     39-1     39-1     39-1
                39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1     39-1
                40-1     40-1     40-1     40-1     40-1     40-1     40-1     40-1     40-1     40-1     40-1     45-1
$SWREG           6-0#    10-24
$SWRMK           4-476    4-476    4-476    4-476    4-476    4-476    4-476    4-476    4-476    39-1     39-1     39-1     39-1     39-1
                39-1     39-1     39-1     39-1     39-1
$TESTN           6-0#    13-2*    13-33*   13-105*  13-210*  13-252*  13-277*  13-328*  13-346*  13-448*  13-473*  13-490*  13-514*  13-536*
                13-561*  13-583*  13-613*  13-631*  13-662*  13-698*  13-733*  13-765*  13-830*  13-884*  13-915*  13-959*  13-988*  13-:54*
                13-;21*  13-;87*  13-<53*  13-<95*  13-=27*  13-=60*  13->11*  13->41*  13->71*  13-?05*  13-?39*  13-?67*  13-@01*  13-@35*
                13-@84*  13-A28*  13-A70*  13-B04*  13-B41*  13-B77*  13-C70*  13-D14*  13-D65*  13-E10*  13-E51*  13-E91*  13-F30*  13-F69*
                39-1*    41-37
$TIMES           6-0#    10-24*   12-46*   13-33*   13-105*  13-490*  13-514*  13-536*  13-561*  13->11*  13->41*  13->71*  13-?05*  13-?39*
                14-20*   39-1     39-1     39-1     39-1*    39-1*
$TKB             6-0#    38-1     38-1     42-1     42-1     42-1     42-1     42-1     42-1     42-1
```

CROSS REFERENCE TABLE (CREF V04.00)

```
$TKCNT  42-1    42-1    42-1#   42-1*   42-1*   42-1*
$TKINT  11-4    12-47   13-134  42-1    42-1    42-1#
$TKQEN  42-1    42-1    42-1#
$TKQIN  42-1    42-1    42-1#   42-1*   42-1*   42-1*   42-1*
$TKQOU  42-1    42-1    42-1#   42-1*   42-1*   42-1*
$TKQSR  42-1    42-1    42-1    42-1#
$TKS     6-0#   38-1    38-1    42-1    42-1    42-1    42-1    42-1    42-1*   42-1*   42-1*   42-1*   42-1*   42-1*
$TKSRV  42-1    42-1#
$TMP0    6-0#   13-55*  13-64*  13-72*  13-78*  13-80   13-=53* 13-=86* 17-41*  18-35*  25-29*  25-31   56-3    56-6
        56-10
$TMP1    6-0#   11-13*  11-14   11-16   11-36*  11-38   11-42   11-50*  11-52   11-56   11-83*  11-84   11-90   11-92
        11-93   11-95   11-100* 11-101  11-104  11-105  11-107  11-112  13-56*  13-57*  13-65*  13-66*  13-70*  13-71*
        13-73   13-79*  13-85   17-42*  17-43   18-36*  18-37   24-24*  24-25*  24-37   24-39   25-28*  25-37   56-3
        56-5    56-10
$TMP2    6-0#   24-27*  24-28*  24-51   24-54   56-5
$TMP3    6-0#
$TMP4    6-0#
$TN      4-465#  4-475   13-2    13-2    13-2    13-2#   13-33   13-33   13-33   13-33#  13-105  13-105  13-105  13-105#
        13-210  13-210  13-210  13-210# 13-252  13-252  13-252  13-252# 13-277  13-277  13-277  13-277# 13-328  13-328
        13-328  13-328# 13-346  13-346  13-346  13-346# 13-448  13-448  13-448  13-448# 13-473  13-473  13-473  13-473#
        13-490  13-490  13-490  13-490# 13-514  13-514  13-514  13-514# 13-536  13-536  13-536  13-536# 13-561  13-561
        13-561  13-561# 13-583  13-583  13-583  13-583# 13-613  13-613  13-613  13-613# 13-631  13-631  13-631  13-631#
        13-662  13-662  13-662  13-662# 13-698  13-698  13-698  13-698# 13-733  13-733  13-733  13-733# 13-765  13-765
        13-765  13-765# 13-830  13-830  13-830  13-830# 13-884  13-884  13-884  13-884# 13-915  13-915  13-915  13-915#
        13-959  13-959  13-959  13-959# 13-988  13-988  13-988  13-988# 13-:54  13-:54  13-:54  13-:54# 13-:21  13-:21
        13-:21  13-:21# 13-:87  13-:87  13-:87  13-:87# 13-<53  13-<53  13-<53  13-<53# 13-<95  13-<95  13-<95  13-<95#
        13-=27  13-=27  13-=27  13-=27# 13-=60  13-=60  13-=60  13-=60# 13->11  13->11  13->11  13->11# 13->41  13->41
        13->41  13->41# 13->71  13->71  13->71  13->71# 13-?05  13-?05  13-?05  13-?05# 13-?39  13-?39  13-?39  13-?39#
        13-?67  13-?67  13-?67  13-?67# 13-@01  13-@01  13-@01  13-@01# 13-@35  13-@35  13-@35  13-@35# 13-@84  13-@84
        13-@84  13-@84# 13-A28  13-A28  13-A28  13-A28# 13-A70  13-A70  13-A70  13-A70# 13-B04  13-B04  13-B04  13-B04#
        13-B41  13-B41  13-B41  13-B41# 13-B77  13-B77  13-B77  13-B77# 13-C70  13-C70  13-C70  13-C70# 13-D14  13-D14
        13-D14  13-D14# 13-D65  13-D65  13-D65  13-D65# 13-E10  13-E10  13-E10  13-E10# 13-E51  13-E51  13-E51  13-E51#
        13-E91  13-E91  13-E91  13-E91# 13-F30  13-F30  13-F30  13-F30# 13-F69  13-F69  13-F69  13-F69# 39-1   39-1
$TPB     6-0#   38-1    38-1    38-1*
$TPFLG   6-0#   38-1    38-1    38-1
$TPS     6-0#   38-1    38-1    38-1
$TRAP   10-24   44-1#
$TRAP2  44-1    44-1#
$TRP    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1
        44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1
        44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1
        44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1    44-1#   44-1#   44-1#   44-1#
        44-1#   44-1#   44-1#   44-1#   44-1#   44-1#   44-1#   44-1#   44-1#   44-1#
$TRPAD  44-1    44-1#
$TSTM    5-8#
$TSTNM   6-0#   12-45*  14-20*  39-1    39-1    39-1    39-1    39-1    39-1*   39-1*   40-1    40-1    40-1
$TTYIN  42-1    42-1    42-1    42-1    42-1    42-1#
$TYPBN  35-1#   44-1    44-1
$TYPDS  36-1#   44-1    44-1
$TYPE   38-1#   44-1    44-1    46-1
$TYPEC  38-1    38-1    38-1    38-1#   42-1
$TYPEX  38-1    38-1    38-1#
$TYPOC  37-1#   44-1    44-1
$TYPON  37-1    37-1#   44-1
$TYPOS  37-1#   44-1
$UNIT    6-0#   12-49*  12-54   12-60   13-40   13-43   13-94   21-49   41-22
```

CROSS REFERENCE TABLE (CREF V04.00)

```
$UNITM   5-8#
$USWR    6-0#
$VECT1   6-0#    11-46    11-56*   13-769   13-775   13-834   13-840
$VECT2   6-0#
$XOFF   38-1     38-1
$XON    38-1     38-1     42-1
$XTSTR  39-1#
.$ASTA  46-1     46-1
.$X      5-8      5-8#
A16      4-715#  13-=96
A17      4-714#  13-=96
ABASE    4-766#   6-0      6-0
ACDW1    6-0      6-0
ACDW2    6-0      6-0
ACKSTS  13-466   15-151   29-15#
ACPUOP   6-0      6-0
ADDW0    6-0      6-0
ADDW1    6-0      6-0
ADDW10   6-0
ADDW11   6-0
ADDW12   6-0
ADDW13   6-0
ADDW14   6-0
ADDW15   6-0
ADDW2    6-0      6-0
ADDW3    6-0      6-0
ADDW4    6-0      6-0
ADDW5    6-0      6-0
ADDW6    6-0      6-0
ADDW7    6-0      6-0
ADDW8    6-0
ADDW9    6-0
ADEVCT   6-0      6-0
ADEVM    6-0      6-0
ADR     13-2     13-2#    13-2#    13-33    13-33#   13-33#   13-105   13-105#  13-105#  13-210   13-210#  13-210#  13-252   13-252#
        13-252#  13-277   13-277#  13-277#  13-328   13-328#  13-328#  13-346   13-346#  13-346#  13-448   13-448#  13-448#  13-473
        13-473#  13-473#  13-490   13-490#  13-490#  13-514   13-514#  13-514#  13-536   13-536#  13-536#  13-561   13-561#  13-561#
        13-583   13-583#  13-583#  13-613   13-613#  13-613#  13-631   13-631#  13-631#  13-662   13-662#  13-662#  13-698   13-698#
        13-698#  13-733   13-733#  13-733#  13-765   13-765#  13-765#  13-830   13-830#  13-830#  13-884   13-884#  13-884#  13-915
        13-915#  13-915#  13-959   13-959#  13-959#  13-988   13-988#  13-988#  13-:54   13-:54#  13-:54#  13-:21   13-:21#  13-:21#
        13-;87   13-;87#  13-;87#  13-<53   13-<53#  13-<53#  13-<95   13-<95#  13-<95#  13-=27   13-=27#  13-=27#  13-=60   13-=60#
        13-=60#  13->11   13->11#  13->11#  13->41   13->41#  13->41#  13->71   13->71#  13->71#  13-?05   13-?05#  13-?05#  13-?39
        13-?39#  13-?39#  13-?67   13-?67#  13-?67#  13-@01   13-@01#  13-@01#  13-@35   13-@35#  13-@35#  13-@84   13-@84#  13-@84#
        13-A28   13-A28#  13-A28#  13-A70   13-A70#  13-A70#  13-B04   13-B04#  13-B04#  13-B41   13-B41#  13-B41#  13-B77   13-B77#
        13-B77#  13-C70   13-C70#  13-C70#  13-D14   13-D14#  13-D14#  13-D65   13-D65#  13-D65#  13-E10   13-E10#  13-E10#  13-E51
        13-E51#  13-E51#  13-E91   13-E91#  13-E91#  13-F30   13-F30#  13-F30#  13-F69   13-F69#  13-F69#
AENV     6-0      6-0
AENVM    6-0      6-0
AFATAL   6-0      6-0
ALL     11-86    47-4#
AMADR1   6-0      6-0
AMADR2   6-0      6-0
AMADR3   6-0      6-0
AMADR4   6-0      6-0
AMAMS1   6-0      6-0
AMAMS2   6-0      6-0
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
AUNIT    6-0       6-0
AUSWR    6-0       6-0
AVECT1   4-767#    6-0       6-0
AVECT2   6-0       6-0
BACK     13-114    13-114#   13-132    13-132#   13-138    13-138#   13-219    13-219#   13-236    13-236#   13-243    13-243#   13-245    13-245#
         13-245#   13-249    13-249#   13-249#   13-262    13-262#   13-283    13-283#   13-337    13-337#   13-340    13-340#   13-340#   13-343
         13-343#   13-343#   13-353    13-353#   13-393    13-393#   13-439    13-439#   13-460    13-460#   13-463    13-463#   13-463#   13-466
         13-466#   13-466#   13-470    13-470#   13-470#   13-475    13-475#   13-481    13-481#   13-483    13-483#   13-483#   13-485    13-485#
         13-485#   13-487    13-487#   13-487#   13-492    13-492#   13-499    13-499#   13-505    13-505#   13-507    13-507#   13-507#   13-509
         13-509#   13-509#   13-511    13-511#   13-511#   13-516    13-516#   13-526    13-526#   13-528    13-528#   13-528#   13-533    13-533#
         13-533#   13-538    13-538#   13-548    13-548#   13-552    13-552#   13-554    13-554#   13-554#   13-556    13-556#   13-556#   13-558
         13-558#   13-558#   13-563    13-563#   13-570#   13-572#   13-574    13-574#   13-576    13-576#   13-576#   13-578    13-578#   13-578#
         13-580    13-580#   13-580#   13-585    13-585#   13-598    13-598#   13-603    13-603#   13-606    13-606#   13-606#   13-608    13-608#
         13-608#   13-610    13-610#   13-610#   13-615    13-615#   13-620    13-620#   13-622    13-622#   13-622#   13-624    13-624#   13-624#
         13-626    13-626#   13-626#   13-628    13-628#   13-628#   13-633    13-633#   13-639    13-639#   13-641    13-641#   13-641#   13-655
         13-655#   13-655#   13-657    13-657#   13-657#   13-659    13-659#   13-659#   13-664    13-664#   13-669    13-669#   13-677    13-677#
         13-677#   13-681    13-681#   13-683    13-683#   13-683#   13-691    13-691#   13-691#   13-693    13-693#   13-693#   13-695    13-695#
         13-695#   13-700    13-700#   13-705    13-705#   13-718    13-718#   13-720    13-720#   13-722    13-722#   13-722#   13-730    13-730#
         13-730#   13-735    13-735#   13-740    13-740#   13-752    13-752#   13-760    13-760#   13-760#   13-762    13-762#   13-762#   13-767
         13-767#   13-791    13-791#   13-806    13-806#   13-808    13-808#   13-808#   13-816    13-816#   13-818    13-818#   13-818#   13-820
         13-820#   13-820#   13-822    13-822#   13-822#   13-824    13-824#   13-824#   13-832    13-832#   13-877    13-877#   13-886    13-886#
         13-892    13-892#   13-894    13-894#   13-894#   13-908    13-908#   13-908#   13-910    13-910#   13-910#   13-912    13-912#   13-912#
         13-917    13-917#   13-929    13-929#   13-933    13-933#   13-935    13-935#   13-935#   13-952    13-952#   13-952#   13-954    13-954#
         13-954#   13-956    13-956#   13-956#   13-961    13-961#   13-966    13-966#   13-968    13-968#   13-968#   13-979    13-979#   13-990
         13-990#   13-:23    13-:23#   13-:26    13-:26#   13-:28    13-:28#   13-:28#   13-:56    13-:56#   13-:90    13-:90#   13-:93    13-:93#
         13-:95    13-:95#   13-:95#   13-;23    13-;23#   13-;56    13-;56#   13-;59    13-;59#   13-;61    13-;61#   13-;61#   13-;89    13-;89#
         13-<22    13-<22#   13-<25    13-<25#   13-<27    13-<27#   13-<27#   13-<55    13-<55#   13-<58#   13-<61    13-<61#   13-<80    13-<80#
         13-<83#   13-<85    13-<85#   13-<87    13-<87#   13-<87#   13-<97    13-<97#   13-=00#   13-=08    13-=08#   13-=17    13-=17#   13-=29
         13-=29#   13-=33    13-=33#   13-=45    13-=45#   13-=47    13-=47#   13-=62    13-=62#   13-=76    13-=76#   13-=78    13-=78#   13-=80
         13-=80#   13-=80#   13->14    13->14#   13->24    13->24#   13->28    13->28#   13->30    13->30#   13->30#   13->32    13->32#   13->32#
         13->34    13->34#   13->34#   13->36    13->36#   13->36#   13->44    13->44#   13->54    13->54#   13->58    13->58#   13->60    13->60#
         13->60#   13->62    13->62#   13->62#   13->64    13->64#   13->64#   13->66    13->66#   13->66#   13->75    13->75#   13->84    13->84#
         13->89    13->89#   13->91    13->91#   13->91#   13->93    13->93#   13->93#   13->95    13->95#   13->95#   13->97    13->97#   13->97#
         13-?08    13-?08#   13-?20    13-?20#   13-?24    13-?24#   13-?26    13-?26#   13-?26#   13-?28    13-?28#   13-?28#   13-?30    13-?30#
         13-?30#   13-?32    13-?32#   13-?32#   13-?42    13-?42#   13-?52    13-?52#   13-?56    13-?56#   13-?58    13-?58#   13-?58#   13-?60
         13-?60#   13-?60#   13-?62    13-?62#   13-?62#   13-?64    13-?64#   13-?64#   13-?70    13-?70#   13-?81    13-?81#   13-?85    13-?85#
         13-?87    13-?87#   13-?87#   13-?89    13-?89#   13-?89#   13-?91    13-?91#   13-?91#   13-?93    13-?93#   13-?93#   13-@04    13-@04#
         13-@15    13-@15#   13-@19    13-@19#   13-@21    13-@21#   13-@21#   13-@23    13-@23#   13-@23#   13-@25    13-@25#   13-@25#   13-@27
         13-@27#   13-@27#   13-@38    13-@38#   13-@54    13-@54#   13-@56    13-@56#   13-@56#   13-@58    13-@58#   13-@62    13-@62#   13-@64
         13-@64#   13-@64#   13-@66    13-@66#   13-@66#   13-@69    13-@69#   13-@69#   13-@71    13-@71#   13-@71#   13-@86    13-@86#   13-A03
         13-A03#   13-A05    13-A05#   13-A05#   13-A07    13-A07#   13-A11    13-A11#   13-A13    13-A13#   13-A13#   13-A15    13-A15#   13-A15#
         13-A18    13-A18#   13-A18#   13-A20    13-A20#   13-A20#   13-A30    13-A30#   13-A46    13-A46#   13-A48    13-A48#   13-A48#   13-A50
         13-A50#   13-A54    13-A54#   13-A56    13-A56#   13-A56#   13-A58    13-A58#   13-A58#   13-A60    13-A60#   13-A60#   13-A62    13-A62#
         13-A62#   13-A87    13-A87#   13-A89    13-A89#   13-A91    13-A91#   13-A91#   13-A99    13-A99#   13-A99#   13-B01    13-B01#   13-B01#
         13-B06    13-B06#   13-B19    13-B19#   13-B21    13-B21#   13-B30    13-B30#   13-B32    13-B32#   13-B32#   13-B34    13-B34#   13-B34#
         13-B36    13-B36#   13-B36#   13-B38    13-B38#   13-B38#   13-B44    13-B44#   13-B54    13-B54#   13-B61    13-B61#   13-B64#   13-B66
         13-B66#   13-B68    13-B68#   13-B68#   13-B70    13-B70#   13-B70#   13-B72    13-B72#   13-B72#   13-B74    13-B74#   13-B74#   13-B93
         13-B93#   13-C04    13-C04#   13-C08    13-C08#   13-C10    13-C10#   13-C10#   13-C72    13-C72#   13-C86    13-C86#   13-C89    13-C89#
         13-C91    13-C91#   13-C91#   13-C93    13-C93#   13-C93#   13-C95    13-C95#   13-C99    13-C99#   13-D01    13-D01#   13-D01#   13-D03
         13-D03#   13-D03#   13-D05    13-D05#   13-D05#   13-D07    13-D07#   13-D07#   13-D16    13-D16#   13-D31    13-D31#   13-D35    13-D35#
         13-D37    13-D37#   13-D37#   13-D39    13-D39#   13-D39#   13-D43    13-D43#   13-D47    13-D47#   13-D49    13-D49#   13-D49#   13-D51
         13-D51#   13-D51#   13-D53    13-D53#   13-D53#   13-D55    13-D55#   13-D55#   13-D67    13-D67#   13-D83    13-D83#   13-D87    13-D87#
         13-D89    13-D89#   13-D89#   13-D91    13-D91#   13-D91#   13-D93    13-D93#   13-D93#   13-D95    13-D95#   13-D95#   13-D97    13-D97#
         13-D97#   13-E12    13-E12#   13-E29    13-E29#   13-E33    13-E33#   13-E35    13-E35#   13-E35#   13-E37    13-E37#   13-E37#   13-E39
         13-E39#   13-E39#   13-E41    13-E41#   13-E41#   13-E43    13-E43#   13-E43#   13-E53    13-E53#   13-E69    13-E69#   13-E73    13-E73#
         13-E75    13-E75#   13-E75#   13-E77    13-E77#   13-E77#   13-E79    13-E79#   13-E79#   13-E81    13-E81#   13-E81#   13-E83    13-E83#
```

CROSS REFERENCE TABLE (CREF V04.00)

	13-E83#	13-E93	13-E93#	13-F10	13-F10#	13-F14	13-F14#	13-F16	13-F16#	13-F16#	13-F25	13-F25#	13-F25#	13-F27
	13-F27#	13-F27#	13-F32	13-F32#	13-F45	13-F45#	13-F47	13-F47#	13-F58	13-F58#	13-F60	13-F60#	13-F60#	13-F62
	13-F62#	13-F62#	13-F64	13-F64#	13-F64#	13-F66	13-F66#	13-F66#	13-F71	13-F71#	13-F83	13-F83#	13-F91	13-F91#
	13-F95	13-F95#	13-F97	13-F97#	13-F97#	13-F99	13-F99#	13-F99#	13-G01	13-G01#	13-G01#	13-G03	13-G03#	13-G03#
BADTMO	10-3#	10-26												
BAI	4-733#	23-28												
BB00	4-662#													
BB01	4-662#													
BB02	4-662#													
BB03	4-662#													
BB04	4-662#													
BB05	4-662#													
BB06	4-662#													
BB07	4-662#													
BB08	4-662#													
BB09	4-662#													
BIT0	4-479#	13->00	20-43											
BIT00	4-479	4-479#	4-491	4-538	4-556	4-575	4-612	4-630	4-662	4-736	4-754	39-1	39-1	40-1
	40-1													
BIT01	4-479	4-479#	4-490	4-537	4-574	4-611	4-629	4-662	4-736	4-753				
BIT02	4-479	4-479#	4-489	4-536	4-573	4-610	4-628	4-662	4-736	4-752				
BIT03	4-479	4-479#	4-488	4-535	4-572	4-609	4-627	4-662	4-674	4-733	4-751			
BIT04	4-479	4-479#	4-487	4-534	4-571	4-626	4-662	4-732						
BIT05	4-479	4-479#	4-486	4-570	4-608	4-625	4-662	4-673	4-731					
BIT06	4-479	4-479#	4-555	4-569	4-591	4-607	4-624	4-662	4-717	4-730	4-750			
BIT07	4-479	4-479#	4-554	4-568	4-590	4-606	4-623	4-645	4-662	4-672	4-716	4-729		
BIT08	4-479	4-479#	4-531	4-553	4-567	4-589	4-605	4-622	4-662	4-715	4-728	39-1		
BIT09	4-479	4-479#	4-530	4-552	4-566	4-588	4-604	4-621	4-644	4-662	4-714	4-727	39-1	39-1
	40-1	40-1												
BIT1	4-479#	13->00	23-48											
BIT10	4-479#	4-529	4-551	4-565	4-587	4-603	4-620	4-643	4-659	4-671	4-713	4-726	4-749	15-165
	40-1													
BIT11	4-479#	4-485	4-528	4-550	4-564	4-602	4-619	4-637	4-642	4-658	4-670	4-725	4-748	15-167
	39-1													
BIT12	4-479#	4-549	4-563	4-601	4-618	4-641	4-657	4-669	4-724	4-747	15-111			
BIT13	4-479#	4-548	4-562	4-600	4-617	4-636	4-656	4-668	4-680	4-712	4-723	4-746	40-1	
BIT14	4-479#	4-547	4-561	4-599	4-616	4-635	4-655	4-667	4-679	4-711	4-722	4-745	15-73	39-1
BIT15	4-479#	4-546	4-560	4-598	4-615	4-634	4-654	4-666	4-678	4-710	4-721	4-744		
BIT2	4-479#	20-43												
BIT3	4-479#	15-198												
BIT4	4-479#	15-140												
BIT5	4-479#													
BIT6	4-479#	15-87												
BIT7	4-479#	10-71	13-B97	20-20										
BIT8	4-479#													
BIT9	4-479#													
BLNKS1	47-41#													
BLNKS2	10-121	11-34	11-48	47-40#										
BLNKS3	47-39#													
BLNKS4	10-82	47-38#												
BOTADR	41-73*	41-91*	41-94	41-109	41-153#									
BOTFLG	41-54*	41-101*	41-104	41-107*	41-154#									
BPTVEC	4-479#													
BSE	4-666#	22-431												
BUFFER	13-C12	13-C34	58-1#	58-5										
BUFONE	13-:10	13-:77	13-;43	13-<09	13-<67	58-2#								

```
BUFTWO   58-3#
CC        4-658#
CH        4-659#
CHGADR    7-0#    10-15*  10-18*  11-25   11-27*
CHRCNT   41-55*   41-78*  41-84*  41-85   41-88*  41-92   41-97*  41-108* 41-155#
CKSWR    39-1     40-1    40-1    44-1#
CLKADR    7-0#    13-B97  19-6*   19-8    19-12*  19-14   20-20   20-39
CLKVCT    7-0#    19-7*   19-13*
CLR       4-731#  10-84   12-53   13-61   27-16
CLRSTS   13-245   13-@56  13-A05  13-A48  15-98   28-22#
CMNSTA   10-130   12-2#
CMPERR   13-624   13-655  13-677  13-691  13-730  13-820  13-908  13-952  13->34  13->64  13->95  13-?30  13-?62  13-?91
         13-@25   13-@69  13-A18  13-A60  13-A99  13-B36  13-B72  13-D05  13-D53  13-D95  13-E41  13-E81  13-F25  13-F64
         13-G01   24-21#
CNSL01   11-32    47-8#
CNSL02   11-40    47-9#
CNSL03   11-44    47-10#
CNSL04   11-54    47-11#
CNSL07   11-78    47-12#
CNSL08   11-109   47-14#
CNSL09   11-72    47-15#
CNTCLR   13-107   13-212  13-240  13-254  13-279  13-330  13-348  13-450  13-995  13-:61  13-;28  13-;94  13-<61  13-=33
         13-=72   13-@52  13-A01  13-A44  13-A72  13-B79  13-D81  13-E27  13-E67  15-76   27-12#
COMMA    11-103   12-6    12-14   47-6#
CONT      4-624#
CPSAVE   39-1     39-1    39-1    39-1*   39-1*   40-1    40-1    40-1    40-1    40-1    40-1#   40-1*   40-1*   41-172
CR        4-479#  11-90   11-101  38-1    38-1    41-76   54-79
CRLF      4-479#  10-6    10-29   10-29   10-43   10-54   10-60   12-30   38-1    38-1    47-4    47-7    47-8    47-9
         47-11    47-12   47-13   47-14   47-15   47-16   47-18   47-26   47-28   47-29   47-30   47-31   47-31   47-34
         47-35    55-7    55-11   55-12   55-15   55-16   55-18   58-8    58-9    58-10   58-11   58-12   58-13   58-14
         58-15    58-16   58-17   58-18   58-19   58-20   58-21   58-22   58-23   58-24   58-25   58-26   58-27   58-28
         58-29    58-30   58-31   58-32   58-33   58-34   58-35   58-36   58-37   58-38   58-39   58-40   58-41   58-42
         58-43    58-44   58-45   58-46   58-47   58-48   58-49   58-50   58-51   58-52   58-53   58-54   58-55   58-56
         58-57    58-58   58-59   58-60   58-61   58-62   58-63   58-64   58-65   58-66   58-67   58-68   58-69   58-70
         58-71    58-72   58-73   58-74   58-75   58-76   58-77   58-78   58-79   58-80   58-81   58-82   58-83   58-84
         58-85    58-86   58-87
CYLMSK    4-649#
DBCK      4-598#
DBEN      4-599#  13-120  13-225  13-=11
DBL       4-749#
DCK       4-560#   4-577  22-497  22-517
DDISP     4-479#   6-0    10-24
DEBL      4-600#
DEVSEL   13-=76   15-62   25-12#
DISPLA    6-0#    10-24*  10-24*  39-1*   40-1*
DISPRE    5-1#    10-24
DLT       4-721#  22-313
DMD       4-612#   4-630# 13-120  13-225  13-350  13-354  13-356  13-391  13-394  13-396  13-452  13-518  13-:00  13-:02
         13-:66   13-:68  13-;33  13-;35  13-;99  13-<01  13-=02  13-=04  13-=11  13-A74  13-E95
DONE     13-92    47-35#
DPE       4-674#  13-624  13-655  13-691  13-820  13-908  13-952  13->34  13->64  13->95  13-?30  13-?62  13-?91  13-@25
         13-@69   13-A18  13-A60  13-A99  13-B36  13-B72  13-D05  13-D53  13-D95  13-E41  13-E81  13-F25  13-F64  13-G01
         21-137   22-255  26-32   30-31   32-52
DPEHI     4-745#
DPELO     4-746#
DPR       4-553#  13-203  13-204  13-317  26-200  28-125  28-126  31-30
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-157	8-160	8-163	8-166	8-169	8-172	8-175
	8-178	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218
	8-236	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275
	8-278	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317
	8-320	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363
	8-366	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408
	8-411	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-450	8-453	8-456	8-459
	8-504	8-507	8-510	8-519	8-522	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552
	8-556	8-559	8-563	8-566	8-569	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603
	8-606	8-609	8-612	8-615	8-618	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645
	8-648	8-651	8-654	8-657	8-660	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702
	8-705	8-708	8-711	8-714	8-723	53-1#								
EFT110	8-221	53-4#												
EFT111	8-224	8-227	8-462	8-465	53-5#									
EFT114	53-7#													
EFT2	8-426	8-429	8-513	8-516	53-2#									
EFT223	8-447	8-528	53-8#											
EFT336	8-351	8-681	8-687	8-690	53-10#									
EFT337	8-684	53-11#												
EFT344	8-699	53-12#												
EFT353	8-720	53-14#												
EH1	51-1	51-6	55-1#											
EH110	51-4	55-2#												
EH111	51-5	55-3#												
EH114	51-7	55-5#												
EH223	51-8	55-6#												
EH256	51-9	55-7#												
EH336	51-11	55-10#												
EH337	51-12	55-11#												
EH344	51-13	55-15#												
EH353	51-15	55-18#												
EHT1	8-3	8-6	8-9	8-12	8-15	8-18	8-24	8-27	8-30	8-33	8-36	8-39	8-42	8-45
	8-48	8-51	8-54	8-57	8-60	8-63	8-66	8-69	8-72	8-75	8-78	8-81	8-84	8-87
	8-90	8-93	8-96	8-99	8-102	8-105	8-108	8-111	8-114	8-117	8-120	8-123	8-126	8-130
	8-133	8-136	8-139	8-142	8-146	8-150	8-154	8-157	8-160	8-163	8-166	8-169	8-172	8-175
	8-178	8-182	8-185	8-188	8-191	8-194	8-197	8-200	8-203	8-206	8-209	8-212	8-215	8-218
	8-236	8-239	8-242	8-245	8-248	8-251	8-254	8-257	8-260	8-263	8-266	8-269	8-272	8-275
	8-278	8-281	8-284	8-287	8-290	8-293	8-296	8-299	8-302	8-305	8-308	8-311	8-314	8-317
	8-320	8-323	8-326	8-329	8-332	8-335	8-338	8-341	8-344	8-347	8-354	8-357	8-360	8-363
	8-366	8-369	8-372	8-375	8-378	8-381	8-384	8-390	8-393	8-396	8-399	8-402	8-405	8-408
	8-411	8-414	8-417	8-420	8-423	8-432	8-435	8-438	8-441	8-444	8-450	8-453	8-456	8-459
	8-504	8-507	8-510	8-519	8-522	8-525	8-531	8-534	8-537	8-540	8-543	8-546	8-549	8-552
	8-556	8-559	8-563	8-566	8-569	8-573	8-577	8-581	8-586	8-590	8-594	8-597	8-600	8-603
	8-606	8-609	8-612	8-615	8-618	8-621	8-624	8-627	8-630	8-633	8-636	8-639	8-642	8-645
	8-648	8-651	8-654	8-657	8-660	8-663	8-666	8-669	8-672	8-675	8-678	8-693	8-696	8-702
	8-705	8-708	8-711	8-714	8-723	51-1#								
EHT110	8-221	51-4#												
EHT111	8-224	8-227	51-5#											
EHT112	8-462	8-465	51-6#											
EHT114	51-7#													
EHT2	8-426	8-429	8-513	8-516	51-2#									
EHT223	8-447	51-8#												
EHT256	8-528	51-9#												
EHT336	8-351	8-681	8-687	8-690	51-11#									
EHT337	8-684	51-12#												
EHT344	8-699	51-13#												

CROSS REFERENCE TABLE (CREF V04.00)

EHT353 8-720 51-15#
EMS1 50-3 50-154 54-3#
EMS10 50-7 50-11 54-10#
EMS100 50-39 50-71 50-176 50-196 54-66#
EMS101 50-40 50-59 50-174 50-197 54-67#
EMS102 50-177 50-178 50-181 50-182 50-183 50-184 50-186 50-187 50-188 50-189 50-234 50-235 54-68#
EMS103 50-56 50-57 50-135 50-141 50-196 50-197 50-198 50-203 50-204 50-205 50-206 50-207 50-208 50-209
50-210 50-211 50-213 50-214 50-215 50-219 50-220 54-69#
EMS104 50-136 50-180 50-203 54-70#
EMS105 50-204 54-71#
EMS106 50-212 50-232 54-72#
EMS11 50-12 50-15 50-16 50-17 50-18 50-19 50-20 50-21 50-22 50-23 50-24 50-25 50-26 50-27
50-28 50-29 50-30 50-31 50-32 50-33 50-43 50-152 50-177 54-11#
EMS110 50-9 50-74 50-77 54-73#
EMS111 50-74 50-226 54-74#
EMS112 50-74 54-75#
EMS113 50-75 50-76 50-127 54-77#
EMS114 50-77 54-78#
EMS115 50-44 50-45 50-48 50-50 50-51 50-52 50-54 50-55 50-60 50-61 50-63 50-66 50-67 50-70
50-73 50-77 50-107 50-108 50-109 50-110 50-134 50-137 50-139 50-140 50-142 50-150 50-151 50-153
50-179 50-182 50-183 50-185 50-186 50-187 50-188 50-189 50-190 50-192 50-193 50-194 50-195 50-201
50-205 50-221 50-222 50-223 50-232 50-234 54-79#
EMS116 50-74 50-77 50-191 50-225 54-80#
EMS117 50-77 50-143 50-148 50-149 50-150 50-178 50-185 50-201 50-225 50-228 50-229 50-230 50-231 54-81#
EMS12 50-12 54-12#
EMS120 54-82#
EMS121 50-79 50-88 50-97 50-99 54-83#
EMS122 50-80 50-89 54-84#
EMS123 50-81 50-88 50-89 50-90 50-90 50-91 50-92 50-93 50-94 50-95 50-96 50-98 50-126 54-85#
EMS124 50-56 50-82 50-91 50-101 54-86#
EMS125 50-83 50-92 50-102 50-169 54-87#
EMS126 50-84 50-93 50-103 54-88#
EMS127 54-89#
EMS13 50-13 50-42 50-216 54-13#
EMS130 50-86 50-95 50-106 54-90#
EMS131 50-87 50-96 50-104 54-91#
EMS132 50-79 50-80 50-81 50-82 50-83 50-84 50-85 50-86 50-87 50-88 50-89 50-90 50-91 50-92
50-93 50-94 50-95 50-96 50-97 50-98 50-99 50-100 50-101 50-102 50-103 50-104 50-105 50-106
50-125 50-126 50-168 50-169 54-92#
EMS133 50-79 50-80 50-81 50-82 50-83 50-84 50-85 50-86 50-87 50-125 54-93#
EMS134 50-88 50-89 50-90 50-91 50-92 50-93 50-94 50-95 50-96 50-126 54-94#
EMS135 50-97 50-98 54-95#
EMS136 50-99 50-100 50-101 50-102 50-103 50-104 50-105 50-106 54-96#
EMS137 50-85 50-94 50-100 54-97#
EMS14 50-13 54-14#
EMS140 50-44 50-52 50-60 50-107 50-117 50-137 50-186 50-192 54-98#
EMS141 50-35 50-44 50-45 50-60 50-61 50-66 50-107 50-122 50-137 50-187 50-188 50-193 50-222 54-99#
EMS142 50-47 50-65 50-108 50-140 50-184 50-202 54-100#
EMS143 50-47 50-65 50-73 50-108 50-109 50-110 50-133 50-140 50-184 50-202 50-205 54-101#
EMS144 50-36 50-73 50-109 50-110 50-205 54-102#
EMS145 50-36 50-73 50-109 50-110 50-205 54-103#
EMS146 50-109 50-185 50-190 50-201 54-104#
EMS147 50-69 50-112 50-113 50-173 50-217 54-105#
EMS15 50-13 50-42 50-216 54-15#
EMS150 50-50 50-51 50-54 50-63 50-67 50-111 50-134 50-138 50-182 50-183 50-194 50-195 50-223 50-234
54-106#

CROSS REFERENCE TABLE (CREF V04.00)

```
EMS151  50-69   50-112  50-113  50-173  50-217  54-107#
EMS152  50-50   50-51   50-54   50-63   50-111  50-134  50-182  50-183  50-194  50-195  50-223  54-108#
EMS153  50-111  54-109#
EMS154  50-111  50-112  50-123  50-124  54-110#
EMS155  50-112  50-124  54-111#
EMS156  50-113  50-114  50-115  50-116  50-151  50-173  54-112#
EMS157  50-113  50-123  54-113#
EMS16   50-13   54-16#
EMS160  50-114  50-115  50-116  54-114#
EMS161  50-114  54-115#
EMS162  54-116#
EMS163  50-58   50-59   50-60   50-61   50-62   50-63   50-64   50-65   50-66   50-67   50-68   50-69   50-70   50-71
        50-72   50-73   50-117  50-143  54-117#
EMS164  50-45   54-118#
EMS165  50-56   50-57   50-71   50-72   50-172  50-196  50-197  50-198  54-120#
EMS166  50-142  54-121#
EMS167  50-8    50-49   50-68   50-123  50-124  50-168  50-169  50-170  50-175  50-235  54-122#
EMS17   50-14   54-17#
EMS170  50-9    54-123#
EMS171  50-57   50-105  50-125  50-126  54-124#
EMS172  50-215  54-125#
EMS173  50-214  54-126#
EMS174  50-216  54-127#
EMS175  50-216  50-217  50-218  50-219  54-128#
EMS176  50-155  54-129#
EMS177  50-127  54-130#
EMS2    50-4    50-5    54-4#
EMS20   50-34   50-53   50-58   50-110  50-118  50-119  50-120  50-121  50-212  50-232  50-238  54-18#
EMS200  50-128  54-131#
EMS201  50-128  54-132#
EMS202  50-122  50-130  50-131  50-132  50-133  50-134  50-136  50-180  54-133#
EMS203  50-122  50-130  50-131  50-132  50-133  50-155  50-174  50-175  50-176  50-237  54-134#
EMS204  50-131  54-135#
EMS205  50-55   50-142  50-179  50-233  54-136#
EMS206  50-55   50-142  50-170  50-179  50-233  54-137#
EMS207  50-168  50-170  54-138#
EMS21   50-15   54-19#
EMS210  50-169  50-170  50-173  54-139#
EMS211  50-170  54-140#
EMS212  50-171  50-172  54-141#
EMS213  50-171  54-142#
EMS214  50-55   54-143#
EMS215  50-236  54-144#
EMS216  50-237  54-145#
EMS217  50-237  54-146#
EMS22   50-16   54-20#
EMS220  50-179  54-147#
EMS221  50-231  50-238  54-148#
EMS222  50-233  54-149#
EMS223  50-232  54-150#
EMS224  50-121  54-151#
EMS225  50-53   50-150  50-151  50-153  54-152#
EMS226  50-53   50-150  50-153  54-153#
EMS227  50-151  54-154#
EMS23   50-17   54-21#
EMS230  50-151  50-152  50-153  54-155#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
EMS231  50-151  54-156#
EMS232  54-157#
EMS24   50-18   54-22#
EMS25   50-19   50-115  54-23#
EMS26   50-20   50-116  50-173  54-24#
EMS27   50-16   50-19   50-20   50-115  50-116  50-173  54-25#
EMS3    50-4    50-76   54-5#
EMS30   50-17   50-18   50-21   50-22   50-23   50-24   50-25   50-26   50-27   50-28   50-29   50-30   50-31   50-32
        50-33   50-43   50-122  50-130  50-131  50-132  50-133  50-144  50-145  50-146  50-147  50-152  50-155  50-174
        50-175  50-176  50-177  50-199  54-26#
EMS31   50-21   50-146  50-207  50-208  54-27#
EMS32   50-22   54-28#
EMS33   50-23   50-219  54-29#
EMS34   50-24   50-225  50-226  50-227  54-30#
EMS35   50-25   50-216  54-31#
EMS36   50-26   50-213  50-214  50-215  54-32#
EMS37   50-27   54-33#
EMS4    50-5    50-75   54-6#
EMS40   50-28   50-220  54-34#
EMS41   50-29   50-190  50-191  54-35#
EMS42   50-30   50-206  54-36#
EMS43   50-31   50-209  54-37#
EMS44   50-32   50-211  50-230  54-38#
EMS45   50-33   50-210  50-229  54-39#
EMS46   50-43   50-58   50-120  50-145  50-177  50-185  50-199  50-201  54-40#
EMS47   50-37   50-44   50-45   50-52   50-60   50-61   50-66   50-117  50-118  50-147  50-186  50-187  50-188  50-192
        50-193  50-222  54-41#
EMS5    50-6    50-7    54-7#
EMS50   50-46   50-48   50-64   50-70   50-132  50-135  50-139  50-141  50-181  50-185  50-189  50-200  50-201  50-221
        54-42#
EMS51   50-48   50-55   50-70   50-130  50-139  50-142  50-179  50-189  50-221  54-43#
EMS52   50-55   50-142  50-143  50-148  50-149  50-150  50-178  50-179  50-185  50-201  50-228  50-229  50-230  50-231
        50-233  54-44#
EMS53   50-37   50-38   50-39   50-40   50-44   50-45   50-46   50-47   50-62   50-64   50-65   50-73   50-140  50-141
        50-181  50-182  50-184  50-186  50-187  50-190  50-191  50-192  50-193  50-194  50-195  50-200  50-202  50-203
        50-204  50-206  50-207  50-216  50-218  50-219  50-220  50-226  54-45#
EMS54   50-226  54-46#
EMS55   50-227  54-47#
EMS56   50-50   50-51   50-54   50-62   50-63   50-67   50-119  50-143  50-144  50-178  50-182  50-183  50-194  50-195
        50-223  50-228  50-234  54-48#
EMS57   50-151  50-190  50-191  50-192  50-193  50-194  50-195  50-196  50-197  50-198  50-199  50-200  50-201  50-202
        50-203  50-204  50-205  50-206  50-207  50-208  50-209  50-210  50-211  50-213  50-214  50-215  50-220  50-221
        50-222  50-223  50-228  50-229  50-230  54-49#
EMS6    50-6    54-8#
EMS60   50-224  54-50#
EMS61   50-224  54-51#
EMS62   50-224  54-52#
EMS63   50-42   50-149  50-190  54-53#
EMS64   50-8    50-14   50-41   50-144  50-145  50-146  50-147  50-199  54-54#
EMS65   50-41   54-55#
EMS66   50-34   50-218  54-56#
EMS67   50-43   50-44   50-45   50-46   50-47   50-48   50-49   50-50   50-51   50-52   50-54   54-57#
EMS7    50-10   54-9#
EMS70   50-48   50-50   50-52   50-54   50-66   50-70   50-73   50-107  50-110  50-139  50-182  50-189  50-194  50-205
        50-221  54-58#
EMS71   50-51   50-59   50-63   50-183  50-195  50-208  54-59#
```

CROSS REFERENCE TABLE (CREF V04.00)

EMT152 8-323 50-108#
EMT153 8-326 50-109#
EMT154 8-329 50-110#
EMT155 8-332 50-111#
EMT156 8-335 50-112#
EMT157 8-338 50-113#
EMT16 8-42 50-16#
EMT160 8-341 50-114#
EMT161 8-344 50-115#
EMT162 8-347 50-116#
EMT163 50-117#
EMT164 8-354 50-118#
EMT165 8-357 50-119#
EMT166 8-360 50-120#
EMT167 8-363 50-121#
EMT17 8-45 50-17#
EMT170 8-366 50-122#
EMT171 8-369 50-123#
EMT172 8-372 50-124#
EMT173 8-375 50-125#
EMT174 8-378 50-126#
EMT175 8-381 50-127#
EMT176 8-384 50-128#
EMT177 50-129#
EMT2 8-6 50-4#
EMT20 8-48 50-18#
EMT200 8-390 50-130#
EMT201 8-393 50-131#
EMT202 8-396 50-132#
EMT203 8-399 50-133#
EMT204 8-402 50-134#
EMT205 8-405 50-135#
EMT206 8-408 50-136#
EMT207 8-411 50-137#
EMT21 8-51 50-19#
EMT210 8-414 50-138#
EMT211 8-417 50-139#
EMT212 8-420 50-140#
EMT213 8-423 50-141#
EMT214 8-426 50-142#
EMT215 8-429 50-143#
EMT216 8-432 50-144#
EMT217 8-435 50-145#
EMT22 8-54 50-20#
EMT220 8-438 50-146#
EMT221 8-441 50-147#
EMT222 8-444 50-148#
EMT223 8-447 50-149#
EMT224 8-450 50-150#
EMT225 8-453 50-151#
EMT226 8-456 50-152#
EMT227 8-459 50-153#
EMT23 8-57 50-21#
EMT230 8-462 50-154#
EMT231 8-465 50-155#
EMT232 8-468 50-156#

CROSS REFERENCE TABLE (CREF V04.00)

EMT233	8-471	50-157#
EMT234	50-158#	
EMT235	50-159#	
EMT236	50-160#	
EMT237	50-161#	
EMT24	8-60	50-22#
EMT240	50-162#	
EMT241	50-163#	
EMT242	50-164#	
EMT243	50-165#	
EMT244	50-166#	
EMT245	50-167#	
EMT246	8-504	50-168#
EMT247	8-507	50-169#
EMT25	8-63	50-23#
EMT250	8-510	50-170#
EMT251	8-513	50-171#
EMT252	8-516	50-172#
EMT253	8-519	50-173#
EMT254	8-522	50-174#
EMT255	8-525	50-175#
EMT256	8-528	50-176#
EMT257	8-531	50-177#
EMT26	8-66	50-24#
EMT260	8-534	50-178#
EMT261	8-537	50-179#
EMT262	8-540	50-180#
EMT263	8-543	50-181#
EMT264	8-546	50-182#
EMT265	8-549	50-183#
EMT266	8-552	50-184#
EMT267	8-556	50-185#
EMT27	8-69	50-25#
EMT270	8-559	50-186#
EMT271	8-563	50-187#
EMT272	8-566	50-188#
EMT273	8-569	50-189#
EMT274	8-573	50-190#
EMT275	8-577	50-191#
EMT276	8-581	50-192#
EMT277	8-586	50-193#
EMT3	8-9	50-5#
EMT30	8-72	50-26#
EMT300	8-590	50-194#
EMT301	8-594	50-195#
EMT302	8-597	50-196#
EMT303	8-600	50-197#
EMT304	8-603	50-198#
EMT305	8-606	50-199#
EMT306	8-609	50-200#
EMT307	8-612	50-201#
EMT31	8-75	50-27#
EMT310	8-615	50-202#
EMT311	8-618	50-203#
EMT312	8-621	50-204#
EMT313	8-624	50-205#

CROSS REFERENCE TABLE (CREF V04.00)

EMT314	8-627	50-206#	
EMT315	8-630	50-207#	
EMT316	8-633	50-208#	
EMT317	8-636	50-209#	
EMT32	8-78	50-28#	
EMT320	8-639	50-210#	
EMT321	8-642	50-211#	
EMT322	8-645	50-212#	
EMT323	8-648	50-213#	
EMT324	8-651	50-214#	
EMT325	8-654	50-215#	
EMT326	8-657	50-216#	
EMT327	8-660	50-217#	
EMT33	8-81	50-29#	
EMT330	8-663	50-218#	
EMT331	8-666	50-219#	
EMT332	8-669	50-220#	
EMT333	8-672	50-221#	
EMT334	8-675	50-222#	
EMT335	8-678	50-223#	
EMT336	8-351	8-681	50-224#
EMT337	8-684	50-225#	
EMT34	8-84	50-30#	
EMT340	8-687	50-226#	
EMT341	8-690	50-227#	
EMT342	8-693	50-228#	
EMT343	8-696	50-229#	
EMT344	8-699	50-230#	
EMT345	8-702	50-231#	
EMT346	8-705	50-232#	
EMT347	8-708	50-233#	
EMT35	8-87	50-31#	
EMT350	8-711	50-234#	
EMT351	8-714	50-235#	
EMT352	8-717	50-236#	
EMT353	8-720	50-237#	
EMT354	8-723	50-238#	
EMT36	8-90	50-32#	
EMT37	8-93	50-33#	
EMT4	8-12	50-6#	
EMT40	8-96	50-34#	
EMT41	8-99	50-35#	
EMT42	8-102	50-36#	
EMT43	8-105	50-37#	
EMT44	8-108	50-38#	
EMT45	8-111	50-39#	
EMT46	8-114	50-40#	
EMT47	8-117	50-41#	
EMT5	8-15	50-7#	
EMT50	8-120	50-42#	
EMT51	8-123	50-43#	
EMT52	8-126	50-44#	
EMT53	8-130	50-45#	
EMT54	8-133	50-46#	
EMT55	8-136	50-47#	
EMT56	8-139	50-48#	

CROSS REFERENCE TABLE (CREF V04.00)

```
EMT57     8-142    50-49#
EMT6      8-18     50-8#
EMT60     8-146    50-50#
EMT61     8-150    50-51#
EMT62     8-154    50-52#
EMT63     8-157    50-53#
EMT64     8-160    50-54#
EMT65     8-163    50-55#
EMT66     8-166    50-56#
EMT67     8-169    50-57#
EMT7      8-21     50-9#
EMT70     8-172    50-58#
EMT71     8-175    50-59#
EMT72     8-178    50-60#
EMT73     8-182    50-61#
EMT74     8-185    50-62#
EMT75     8-188    50-63#
EMT76     8-191    50-64#
EMT77     8-194    50-65#
EMTVEC     4-479#  10-24*   10-24*
EQUALS    47-3#
ERR        4-547#  13-288   13-:38   13-;05   13-;71   13-<37   22-84    22-86    22-102   22-136
ERRNMB    41-40*   41-41*   41-44    41-49    41-56    41-152#
ERROR      4-479#
ERRTYP    40-1     41-13#
ERRVEC     4-479#  10-24    10-24*   10-24*   10-26*   10-27*   13-5     13-5     13-6*    13-7*    13-20*   13-20*   13-24*   13-24*
          13-=66   13-=67   13-=68*  13-=69*  13-=08*  13-=09*  13-C66*  13-C67*  17-35    17-35    17-39*   17-40*   17-70*   17-70*
          18-29    18-29    18-33*   18-34*   18-63*   18-63*   19-3     19-3     19-4*    19-5*    19-11*   19-18*   19-18*   20-9
          20-9     20-10*   20-11*   20-31*   20-31*   25-20    25-20    25-21*   25-22*   25-51*   25-51*   27-12    27-12    27-13*
          27-14*   27-25*   27-25*   39-1     39-1     39-1*    39-1*    39-1*    39-1*    39-1*    40-1     40-1*    40-1*
ERTY00    41-21    41-157#
ERTY01    41-38    41-158#
ERTY02    41-43    41-159#
ERTY03    41-45    41-160#
ERTY04    41-144   41-161#
ERTY05    41-31    41-162#
ESRC       4-619#
F0         4-490#  22-29
F1         4-489#  22-29
F2         4-488#  22-29    23-147
F3         4-487#  22-29
F4         4-486#  22-29
FER        4-571#   4-577   22-416
FIND      13-114   13-114#  13-114#  13-132   13-132#  13-132#  13-138   13-138#  13-138#  13-219   13-219#  13-219#  13-236   13-236#
          13-236#  13-243   13-243#  13-243#  13-245   13-245#  13-245#  13-249   13-249#  13-249#  13-262   13-262#  13-262#  13-283
          13-283#  13-283#  13-337   13-337#  13-337#  13-340   13-340#  13-340#  13-343   13-343#  13-343#  13-353   13-353#  13-353#
          13-393   13-393#  13-393#  13-439   13-439#  13-439#  13-460   13-460#  13-460#  13-463   13-463#  13-463#  13-466   13-466#
          13-466#  13-470   13-470#  13-470#  13-475   13-475#  13-475#  13-481   13-481#  13-481#  13-483   13-483#  13-483#  13-485
          13-485#  13-485#  13-487   13-487#  13-487#  13-492   13-492#  13-492#  13-499   13-499#  13-499#  13-505   13-505#  13-505#
          13-507   13-507#  13-507#  13-509   13-509#  13-509#  13-511   13-511#  13-511#  13-516   13-516#  13-516#  13-526   13-526#
          13-526#  13-528   13-528#  13-528#  13-533   13-533#  13-533#  13-538   13-538#  13-538#  13-548   13-548#  13-548#  13-552
          13-552#  13-552#  13-554   13-554#  13-554#  13-556   13-556#  13-556#  13-558   13-558#  13-558#  13-563   13-563#  13-563#
          13-570   13-570#  13-572   13-572#  13-574   13-574#  13-574#  13-576   13-576#  13-576#  13-578   13-578#  13-578#  13-580
          13-580#  13-580#  13-585   13-585#  13-585#  13-598   13-598#  13-598#  13-603   13-603#  13-603#  13-606   13-606#  13-606#
          13-608   13-608#  13-608#  13-610   13-610#  13-610#  13-615   13-615#  13-615#  13-620   13-620#  13-620#  13-622   13-622#
          13-622#  13-624   13-624#  13-624#  13-626   13-626#  13-626#  13-628   13-628#  13-628#  13-633   13-633#  13-633#  13-639
```

13-639#	13-639#	13-641	13-641#	13-641#	13-655	13-655#	13-655#	13-657	13-657#	13-657#	13-659	13-659#	13-659#
13-664	13-664#	13-664#	13-669	13-669#	13-669#	13-677	13-677#	13-677#	13-681	13-681#	13-681#	13-683	13-683#
13-683#	13-691	13-691#	13-691#	13-693	13-693#	13-693#	13-695	13-695#	13-695#	13-700	13-700#	13-700#	13-705
13-705#	13-705#	13-718	13-718#	13-718#	13-720	13-720#	13-720#	13-722	13-722#	13-722#	13-730	13-730#	13-730#
13-735	13-735#	13-735#	13-740	13-740#	13-740#	13-752	13-752#	13-752#	13-760	13-760#	13-760#	13-762	13-762#
13-762#	13-767	13-767#	13-767#	13-791	13-791#	13-791#	13-806	13-806#	13-806#	13-808	13-808#	13-808#	13-816
13-816#	13-816#	13-818	13-818#	13-818#	13-820	13-820#	13-820#	13-822	13-822#	13-822#	13-824	13-824#	13-824#
13-832	13-832#	13-832#	13-877	13-877#	13-877#	13-886	13-886#	13-886#	13-892	13-892#	13-892#	13-894	13-894#
13-894#	13-908	13-908#	13-908#	13-910	13-910#	13-910#	13-912	13-912#	13-912#	13-917	13-917#	13-917#	13-929
13-929#	13-929#	13-933	13-933#	13-933#	13-935	13-935#	13-935#	13-952	13-952#	13-952#	13-954	13-954#	13-954#
13-956	13-956#	13-956#	13-961	13-961#	13-961#	13-966	13-966#	13-966#	13-968	13-968#	13-968#	13-979	13-979#
13-979#	13-990	13-990#	13-990#	13-:23	13-:23#	13-:23#	13-:26	13-:26#	13-:26#	13-:28	13-:28#	13-:28#	13-:56
13-:56#	13-:56#	13-:90	13-:90#	13-:90#	13-:93	13-:93#	13-:93#	13-:95	13-:95#	13-:95#	13-;23	13-;23#	13-;23#
13-;56	13-;56#	13-;56#	13-;59	13-;59#	13-;59#	13-;61	13-;61#	13-;61#	13-;89	13-;89#	13-;89#	13-<22	13-<22#
13-<22#	13-<25	13-<25#	13-<25#	13-<27	13-<27#	13-<27#	13-<55	13-<55#	13-<55#	13-<58	13-<58#	13-<61	13-<61#
13-<61#	13-<80	13-<80#	13-<80#	13-<83	13-<83#	13-<85	13-<85#	13-<85#	13-<87	13-<87#	13-<87#	13-<97	13-<97#
13-<97#	13-=00	13-=00#	13-=08	13-=08#	13-=08#	13-=17	13-=17#	13-=17#	13-=29	13-=29#	13-=29#	13-=33	13-=33#
13-=33#	13-=45	13-=45#	13-=45#	13-=47	13-=47#	13-=47#	13-=62	13-=62#	13-=62#	13-=76	13-=76#	13-=76#	13-=78
13-=78#	13-=78#	13-=80	13-=80#	13-=80#	13->14	13->14#	13->14#	13->24	13->24#	13->24#	13->28	13->28#	13->28#
13->30	13->30#	13->30#	13->32	13->32#	13->32#	13->34	13->34#	13->34#	13->36	13->36#	13->36#	13->44	13->44#
13->44#	13->54	13->54#	13->54#	13->58	13->58#	13->58#	13->60	13->60#	13->60#	13->62	13->62#	13->62#	13->64
13->64#	13->64#	13->66	13->66#	13->66#	13->75	13->75#	13->75#	13->84	13->84#	13->84#	13->89	13->89#	13->89#
13->91	13->91#	13->91#	13->93	13->93#	13->93#	13->95	13->95#	13->95#	13->97	13->97#	13->97#	13-?08	13-?08#
13-?08#	13-?20	13-?20#	13-?20#	13-?24	13-?24#	13-?24#	13-?26	13-?26#	13-?26#	13-?28	13-?28#	13-?28#	13-?30
13-?30#	13-?30#	13-?32	13-?32#	13-?32#	13-?42	13-?42#	13-?42#	13-?52	13-?52#	13-?52#	13-?56	13-?56#	13-?56#
13-?58	13-?58#	13-?58#	13-?60	13-?60#	13-?60#	13-?62	13-?62#	13-?62#	13-?64	13-?64#	13-?64#	13-?70	13-?70#
13-?70#	13-?81	13-?81#	13-?81#	13-?85	13-?85#	13-?85#	13-?87	13-?87#	13-?87#	13-?89	13-?89#	13-?89#	13-?91
13-?91#	13-?91#	13-?93	13-?93#	13-?93#	13-a04	13-a04#	13-a04#	13-a15	13-a15#	13-a15#	13-a19	13-a19#	13-a19#
13-a21	13-a21#	13-a21#	13-a23	13-a23#	13-a23#	13-a25	13-a25#	13-a25#	13-a27	13-a27#	13-a27#	13-a38	13-a38#
13-a38#	13-a54	13-a54#	13-a54#	13-a56	13-a56#	13-a56#	13-a58	13-a58#	13-a58#	13-a62	13-a62#	13-a62#	13-a64
13-a64#	13-a64#	13-a66	13-a66#	13-a66#	13-a69	13-a69#	13-a69#	13-a71	13-a71#	13-a71#	13-a86	13-a86#	13-a86#
13-A03	13-A03#	13-A03#	13-A05	13-A05#	13-A05#	13-A07	13-A07#	13-A07#	13-A11	13-A11#	13-A11#	13-A13	13-A13#
13-A13#	13-A15	13-A15#	13-A15#	13-A18	13-A18#	13-A18#	13-A20	13-A20#	13-A20#	13-A30	13-A30#	13-A30#	13-A46
13-A46#	13-A46#	13-A48	13-A48#	13-A48#	13-A50	13-A50#	13-A50#	13-A54	13-A54#	13-A54#	13-A56	13-A56#	13-A56#
13-A58	13-A58#	13-A58#	13-A60	13-A60#	13-A60#	13-A62	13-A62#	13-A62#	13-A87	13-A87#	13-A87#	13-A89	13-A89#
13-A89#	13-A91	13-A91#	13-A91#	13-A99	13-A99#	13-A99#	13-B01	13-B01#	13-B01#	13-B06	13-B06#	13-B06#	13-B19
13-B19#	13-B19#	13-B21	13-B21#	13-B21#	13-B30	13-B30#	13-B30#	13-B32	13-B32#	13-B32#	13-B34	13-B34#	13-B34#
13-B36	13-B36#	13-B36#	13-B38	13-B38#	13-B38#	13-B44	13-B44#	13-B44#	13-B54	13-B54#	13-B54#	13-B61	13-B61#
13-B61#	13-B64	13-B64#	13-B66	13-B66#	13-B66#	13-B68	13-B68#	13-B68#	13-B70	13-B70#	13-B70#	13-B72	13-B72#
13-B72#	13-B74	13-B74#	13-B74#	13-B93	13-B93#	13-B93#	13-C04	13-C04#	13-C04#	13-C08	13-C08#	13-C08#	13-C10
13-C10#	13-C10#	13-C72	13-C72#	13-C72#	13-C86	13-C86#	13-C86#	13-C89	13-C89#	13-C89#	13-C91	13-C91#	13-C91#
13-C93	13-C93#	13-C93#	13-C95	13-C95#	13-C95#	13-C99	13-C99#	13-C99#	13-D01	13-D01#	13-D01#	13-D03	13-D03#
13-D03#	13-D05	13-D05#	13-D05#	13-D07	13-D07#	13-D07#	13-D16	13-D16#	13-D16#	13-D31	13-D31#	13-D31#	13-D35
13-D35#	13-D35#	13-D37	13-D37#	13-D37#	13-D39	13-D39#	13-D39#	13-D43	13-D43#	13-D43#	13-D47	13-D47#	13-D47#
13-D49	13-D49#	13-D49#	13-D51	13-D51#	13-D51#	13-D53	13-D53#	13-D53#	13-D55	13-D55#	13-D55#	13-D67	13-D67#
13-D67#	13-D83	13-D83#	13-D83#	13-D87	13-D87#	13-D87#	13-D89	13-D89#	13-D89#	13-D91	13-D91#	13-D91#	13-D93
13-D93#	13-D93#	13-D95	13-D95#	13-D95#	13-D97	13-D97#	13-D97#	13-E12	13-E12#	13-E12#	13-E29	13-E29#	13-E29#
13-E33	13-E33#	13-E33#	13-E35	13-E35#	13-E35#	13-E37	13-E37#	13-E37#	13-E39	13-E39#	13-E39#	13-E41	13-E41#
13-E41#	13-E43	13-E43#	13-E43#	13-E53	13-E53#	13-E53#	13-E69	13-E69#	13-E69#	13-E73	13-E73#	13-E73#	13-E75
13-E75#	13-E75#	13-E77	13-E77#	13-E77#	13-E79	13-E79#	13-E79#	13-E81	13-E81#	13-E81#	13-E83	13-E83#	13-E83#
13-E93	13-E93#	13-E93#	13-F10	13-F10#	13-F10#	13-F14	13-F14#	13-F14#	13-F16	13-F16#	13-F16#	13-F25	13-F25#
13-F25#	13-F27	13-F27#	13-F27#	13-F32	13-F32#	13-F32#	13-F45	13-F45#	13-F45#	13-F47	13-F47#	13-F47#	13-F58
13-F58#	13-F58#	13-F60	13-F60#	13-F60#	13-F62	13-F62#	13-F62#	13-F64	13-F64#	13-F64#	13-F66	13-F66#	13-F66#
13-F71	13-F71#	13-F71#	13-F83	13-F83#	13-F83#	13-F91	13-F91#	13-F91#	13-F95	13-F95#	13-F95#	13-F97	13-F97#
13-F97#	13-F99	13-F99#	13-F99#	13-G01	13-G01#	13-G01#	13-G03	13-G03#	13-G03#				
FMT16	4-641#	13-927	13-937	13-:14	13-:81	13-;47	13-<13	13-<71	13-a77	13-a79	13-a92	13-A35	13-C35	13-C55

CROSS REFERENCE TABLE (CREF V04.00)

```
ILRG52   4-702#
ILRG54   4-702#
ILRG56   4-702#
ILRG60   4-702#
ILRG62   4-702#
ILRG64   4-702#
ILRG66   4-702#
ILRG70   4-702#
ILRG72   4-702#
ILRG74   4-702#
ILRG76   4-702#
INSERT  13-48    13-93    47-32#
IOTVEC   4-479#  10-24*   10-24*
IPCK0    4-754#
IPCK1    4-753#
IPCK2    4-752#
IPCK3    4-751#
IR       4-730#  13-156   28-56
IVC      4-669#  13-529   13-:97   13-:99   13-A92   13-A94   13-A96   13-A99   13-F18   13-F20   13-F22   13-F25   22-198   22-199
        26-90    26-213   30-92    30-99    30-102   30-171   32-130   32-154   32-157   32-244   33-58    48-58    48-59    48-60
        48-62    48-63    48-64    48-67    48-68    48-69    48-70    48-71    48-72    48-73    48-74    48-75    48-76    48-77
        48-78    48-79    48-80    48-81    48-82    48-83    48-84    48-85    48-86    48-87    48-88
LBC      4-671#  13-440   13-443   13-444
LBT      4-551#  13-288   22-358   23-125   23-127
LF       4-479#  38-1     38-1     41-80    47-11    47-30    47-31    54-79
LODEV   10-117   47-21#
LS       4-628#  13-179   28-80    31-70
LSC      4-670#  13-591
LST      4-629#  13-179   28-80    31-70
LSTRK    7-0#    12-55*   13-@90   13-E16   23-62    26-54    32-95
MANUAL   7-0#    11-59*   11-69*   13-35
MCLK     4-602#
MCPE     4-712#  13-266   21-90    21-107   21-120   21-123   32-70    32-75
MDF      4-607#  13-391
MDPE     4-728#  13-271   22-482
MI       4-610#
MINSTR  13-41    47-30#
MOC      4-605#  13-391   13-:02   13-:68   13-;35   13-<01   13-=04   13-=11
MOH      4-636#
MOL      4-549#  10-98    13-285   13-289   13-317   13-360   13-363   13-400   13-403   13-404   26-151   26-161   26-162   26-167
        26-172   26-200   29-37    29-40    30-68    30-145   30-146   30-153   30-158   31-30    32-194   32-205   32-206   32-210
        32-213   33-35    33-36    33-40    33-45
MRD      4-603#
MS       4-608#
MSC      4-611#
MSDRVS  11-81    47-16#
MSE      4-678#
MSER     4-606#  13-391
MSGDRV  12-59    47-17#
MSHELP  11-11    47-7#
MSMNUL  11-60    47-28#
MSUNIT  13-39    47-29#
MUR      4-604#  13-391   13-:02   13-:68   13-;35   13-<01   13-=04   13-=11
MWD      4-627#  13-180   28-81    31-71
MWP      4-609#  13-391   13-<01
MXF      4-727#  13-271   22-343
```

CROSS REFERENCE TABLE (CREF V04.00)

```
N        11-19   47-36#
NDTMSK    4-577# 13-624  13-655  13-691  13-820  13-908  13-952  13->34  13->64  13->95  13-?30  13-?62  13-?91  13-@25
         13-@69  13-A18  13-A60  13-A99  13-B36  13-B72  13-D05  13-D53  13-D95  13-E41  13-E81  13-F25  13-F64  13-G01
NED       4-724# 10-87   13-80   13-271  21-72   21-76   25-31
NEM       4-725# 22-328
NONE     12-7    47-27#
NOP       4-495# 13-247  13-617  13-679  13-750  13-863  13-993  13-:59  13-;26  13-;92  13-<59  13-=13
NOTAVL   10-105  47-23#
NOTPRS   10-102  47-22#
NOTRM    10-96   47-20#
NSA       4-634#
OCC       4-615#
OFD       4-645# 13-927  13-937
OFFSET    4-501# 13-565  13-635  13-666  13-702  13-737  13-793  13-854  13-963  13-=15  13-B48  13-F76
OM        4-556# 13-317  13-642  13-645  13-646  13-895  13-899  13-969  13-972  13-973  13-980  13-983  30-145  30-201
         30-204  31-29
OPE       4-668# 13-55   13-57   13-66
OPI       4-562# 22-182  26-145  26-169  29-51   29-69   29-73   30-24   30-61   30-64   30-155  32-187  32-190  32-215
         33-42   48-57   48-58   48-59   48-60   48-61   48-62   48-63   48-64   48-65   48-66   48-67   48-68   48-69
         48-70   48-71   48-72   48-73   48-74   48-75   48-76   48-77   48-78   48-79   48-80   48-81   48-82   48-83
         48-84   48-85   48-86   48-87   48-88
OR        4-729#
PACACK    4-505#
PAKACK    4-504#  4-505  13-332  13-456  13-:06  13-;39  13-<05  13-=06  15-125
PAR       4-572# 13-=49  13-=51  21-135  21-139  21-151  26-30   30-24   30-29   30-34   32-50   32-57
PAT       4-732# 13-=37
PDA       4-622#
PFECH    41-59   41-164#
PFECH1   41-164  41-165#
PFECH2   41-164  41-168#
PFECH3   41-164  41-171#
PFECH4   41-164  41-173#
PGE       4-726# 13-271
PGM       4-552# 13-317  31-29
PHA       4-623#
PIP       4-548# 13-288  13-366  13-369  13-370  13-406  13-409  13-500  13-B22  13-B25  13-F49  13-F53  15-180  26-161
         26-182  26-187  30-145  30-216  30-221  31-29   32-205  32-226  32-231  33-35   33-72   33-77
PIRQ      4-479#
PIRQVE    4-479#
PLFS      4-620#
PR0       4-479#
PR1       4-479#
PR2       4-479#
PR3       4-479# 39-5
PR4       4-479#
PR5       4-479# 12-48  39-9
PR6       4-479# 10-27  13-7    13-774  13-783  13-839  13-842  13-=69  17-40   18-34   19-5    20-11   25-22   27-14
PR7       4-479# 13-781 13-848
PRIERR   13-340  13-463  13-483  13-507  13-528  13-554  13-576  13-606  13-622  13-641  13-683  13-722  13-808  13-818
         13-894  13-935  13-968  13-:28  13-:95  13-;61  13-<27  13-<87  13-=80  13->30  13->60  13->91  13-?26  13-?58
         13-?87  13-@21  13-@64  13-A13  13-A56  13-A91  13-B32  13-B68  13-C10  13-C91  13-D01  13-D37  13-D49  13-D89
         13-E35  13-E75  13-F16  13-F60  13-F97  21-39#
PS        4-479   4-479#
PSEL      4-713#
PSW       4-479#
PULL     13-45   47-33#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
PUT       13-114  13-132  13-219  13-236  13-256  13-332  13-350  13-391  13-437  13-452  13-454  13-456  13-477  13-494
          13-496  13-518  13-520  13-522  13-548  13-565  13-567  13-598  13-617  13-635  13-666  13-679  13-702  13-718
          13-737  13-748  13-750  13-791  13-793  13-854  13-863  13-888  13-929  13-963  13-977  13-:00  13-:02  13-:06
          13-:23  13-:66  13-:68  13-:90  13-;33  13-;35  13-;39  13-;56  13-;99  13-<01  13-<05  13-<22  13-<63  13-<80
          13-=02  13-=04  13-=06  13-=11  13-=13  13-=15  13-=45  13->24  13->54  13->84  13-?20  13-?52  13-?81  13-@15
          13-@58  13-A07  13-A50  13-A74  13-A87  13-B19  13-B54  13-B61  13-B93  13-C04  13-C86  13-C95  13-D31  13-D43
          13-D83  13-E29  13-E69  13-E95  13-F10  13-F45  13-F83  13-F91  15-128  15-186  18-28#
PUTBUF     7-0#   18-31
PUTINX     7-0#   13-111* 13-112* 13-124  13-216* 13-217* 13-228  13-256* 13-256* 13-332* 13-332* 13-350* 13-350* 13-391*
          13-391* 13-437* 13-437* 13-452* 13-452* 13-454* 13-454* 13-456* 13-456* 13-477* 13-477* 13-494* 13-494* 13-496*
          13-496* 13-518* 13-518* 13-520* 13-520* 13-522* 13-522* 13-543  13-565* 13-565* 13-567* 13-567* 13-592  13-617*
          13-617* 13-635* 13-635* 13-666* 13-666* 13-679* 13-679* 13-702* 13-702* 13-716* 13-717* 13-737* 13-737* 13-748*
          13-748* 13-750* 13-750* 13-788* 13-789* 13-793* 13-793* 13-854* 13-854* 13-863* 13-863* 13-888* 13-888* 13-919
          13-963* 13-963* 13-977* 13-977* 13-:00* 13-:00* 13-:02* 13-:02* 13-:06* 13-:06* 13-:15  13-:66* 13-:66* 13-:68*
          13-:68* 13-:82  13-;33* 13-;33* 13-;35* 13-;35* 13-;39* 13-;39* 13-;48  13-;99* 13-;99* 13-<01* 13-<01* 13-<05*
          13-<05* 13-<14  13-<63* 13-<63* 13-<72  13-=02* 13-=02* 13-=04* 13-=04* 13-=06* 13-=06* 13-=11* 13-=11* 13-=13*
          13-=13* 13-=15* 13-=15* 13-=40  13->19  13->49  13->79  13-?14  13-?47  13-?75  13-@09  13-@44  13-@93  13-A36
          13-A74* 13-A74* 13-A80  13-B12  13-B49  13-B59* 13-B60* 13-B85  13-C79  13-D24  13-D73  13-E19  13-E59  13-E95*
          13-E95* 13-F02  13-F38  13-F77  13-F88* 13-F89* 15-126* 15-127* 15-184* 15-185* 18-32  21-142
PWRVEC     4-479# 10-24*  10-24*  45-1*   45-1*   45-1*   45-1*   45-1*   45-1*
QUES      47-5#
R6         4-479# 10-24   10-24*  10-24*
R7         4-479#
RCLSTS    13-485  13-556  13-578  15-209  30-15#
RD         4-520#
RDCHR     11-12   11-82   11-99   42-1    44-1#
RDLIN     11-61   13-51   13-96   43-1    44-1#
RDOCT     11-35   11-49   44-1#
RDY        4-716# 13-143  20-17   20-18   21-86   21-89   28-31
READY     12-37#  14-16   14-20
RECAL      4-498# 13-477  13-496  13-522  13-542  13-567  15-183
RESREG    41-148  44-1#
RESVEC     4-479#
RET       13-50   13-95   47-34#
REX        4-618#
RG         4-616#
RH         4-521#
RIP        4-503# 13-928
RLEASE     4-500#
RMAS       4-688# 13-716  13-788
RMASI      7-0#   13-168  31-50   56-13
RMASO      7-0#   13-715* 13-787*
RMBA       4-760# 13-12*  13-13   13-125  13-229  13-:16  13-:83  13-;49  13-<15  13-<73
RMBAE      4-763#
RMBAEI     7-0#
RMBAEO     7-0#
RMBAI      7-0#   13-149  23-32   23-34   28-42   56-14
RMBAO      7-0#   13-117* 13-222* 13-:10* 13-:77* 13-;43* 13-<09* 13-<67* 23-26  23-30
RMCS1      4-684#  4-758# 10-89   13-9*   13-79   13-129  13-233  13-256  13-258  13-332  13-437  13-456  13-477  13-496
          13-522  13-546  13-565  13-567  13-596  13-617  13-635  13-666  13-679  13-702  13-737  13-750  13-793  13-854
          13-863  13-888  13-923  13-963  13-:06  13-:21  13-:88  13-;39  13-;54  13-<05  13-<20  13-<78  13-=06  13-=13
          13-=15  13-=96* 13->22  13->52  13->82  13-?17  13-?50  13-?78  13-@12  13-@48  13-@97  13-A40  13-A84  13-B16
          13-B52  13-B59  13-B89  13-C83  13-D28  13-D77  13-E23  13-E63  13-F07  13-F42  13-F81  13-F88  15-126  15-184
          17-42   18-36   20-16   25-28
RMCS1I     7-0#   13-141  13-264  21-65   21-67   21-69   21-86   21-88   21-91   21-102  21-106  21-108  21-120  21-122
          21-124  22-51   22-53   22-64   22-107  23-12   28-29   31-16   32-70   32-74   32-76   56-13
```

CZRNDA0 RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE S-26 SEQ 0312
CROSS REFERENCE TABLE (CREF V04.00)

```
RMCS10   7-0#    13-121*  13-226*  13-247*  13-256*  13-332*  13-437*  13-456*  13-477*  13-496*  13-522*  13-542*  13-565*  13-567*
        13-588*  13-617*  13-635*  13-666*  13-679*  13-702*  13-737*  13-750*  13-793*  13-854*  13-863*  13-888*  13-928*  13-963*
        13-:06*  13-:08*  13-:09*  13-:75*  13-:76*  13-;39*  13-;41*  13-;42*  13-<05*  13-<07*  13-<08*  13-<65*  13-<66*  13-=U6*
        13-=13*  13-=15*  13->18*  13->48*  13->78*  13-?13*  13-?46*  13-?74*  13-a08*  13-a40*  13-a88*  13-A32*  13-A76*  13-B11*
        13-B48*  13-B58*  13-B84*  13-C02*  13-C78*  13-C94*  13-D22*  13-D42*  13-D72*  13-E14*  13-E55*  13-F01*  13-F35*  13-F76*
        13-F90*  15-125*  15-183*  22-28    23-48
RMCS2    4-761#  10-84*   10-85*   10-87    12-53*   12-54*   13-14    13-15*   13-16    13-17*   13-54*   13-61*   13-62*   13-70
        13-78    13-130   13-135*  13-234   13-259   13-=41   17-41    18-35    25-27*   25-29    27-16*   27-18*
RMCS2I   7-0#    13-153   13-270   21-47    21-72    21-74    21-75    22-99    22-170   22-271   22-314   22-329   22-344   22-483
        23-28    28-51    56-13
RMCS20   7-0#    13-122*  13-227*  13-=38*
RMCS3    4-764#
RMCS3I   7-0#
RMCS30   7-0#
RMDA     4-685#  13-545   13-593   13-920   13-:19   13-:86   13-;52   13-<18   13-<76   13-=42   13->21   13->51   13->81   13-?15
        13-?49   13-?77   13-a10   13-a46   13-a96   13-A37   13-A82   13-B13   13-B51   13-B86   13-C80   13-D27   13-D76   13-E20
        13-E62   13-F05   13-F40   13-F78
RMDAI    7-0#    13-941   13-943   23-100   23-171   56-14    56-16
RMDAO    7-0#    13-541*  13-587*  13-925*  13-:12*  13-:79*  13-;45*  13-<11*  13-<69*  13-=39*  13->17*  13->47*  13->77*  13-?12*
        13-?45*  13-?73*  13-a07*  13-a43*  13-a73*  13-a74   13-a80*  13-a90*  13-a91*  13-A22*  13-A23   13-A34*  13-A79*  13-B10*
        13-B47*  13-B83*  13-C76*  13-D09*  13-D10   13-D18*  13-D59*  13-D60   13-D71*  13-D99*  13-E00   13-E06*  13-E16*  13-E17*
        13-E45*  13-E46   13-E57*  13-E99*  13-F37*  13-F74*  23-60    23-61    23-100   26-54    26-57    26-61    26-65    32-95
        32-98    32-102   32-106
RMDB     4-762#  13-18*   13-19    13-111   13-216   16-20
RMDBI    7-0#
RMDBO    7-0#    13-110*  13-215*
RMDC     4-694#  13-544   13-921   13-977   13-:20   13-:87   13-;53   13-<19   13-<77   13->20   13->50   13->80   13-?16   13-?48
        13-?76   13-a11   13-a47   13-a95   13-A38   13-A83   13-B14   13-B50   13-B87   13-C81   13-D26   13-D74   13-E21   13-E61
        13-F04   13-F39   13-F79
RMDCI    7-0#    13-946   13-948   23-184   56-14    56-16
RMDCO    7-0#    13-540*  13-926*  13-977*  13-:13*  13-:80*  13-;46*  13-<12*  13-<70*  13->16*  13->46*  13->73*  13->99*  13-?00
        13-?11*  13-?44*  13-?72*  13-?95   13-?97*  13-a06*  13-a29   13-a31*  13-a42*  13-a89*  13-A33*  13-A64*  13-A65   13-A78*
        13-B09*  13-B46*  13-B82*  13-C75*  13-D23*  13-D41*  13-D70*  13-E15*  13-E56*  13-E85*  13-E86   13-E98*  13-F36*  13-F73*
        23-59    26-50    32-91
RMDS     4-686#  10-86    10-98
RMDSI    7-0#    13-202   13-285   13-287   13-290   13-314   13-316   13-318   13-360   13-362   13-366   13-368   13-372   13-374
        13-400   13-402   13-406   13-408   13-412   13-414   13-430   13-432   13-500   13-642   13-644   13-648   13-650   13-670
        13-673   13-684   13-686   13-706   13-708   13-723   13-725   13-741   13-743   13-753   13-756   13-895   13-898   13-901
        13-903   13-969   13-971   13-980   13-982   13-:38   13-:42   13-;05   13-;09   13-;71   13-;75   13-<37   13-<41   13-B22
        13-B24   13-F49   13-F52   15-122   15-180   21-104   22-38    22-40    22-85    22-102   22-136   22-140   22-196   22-223
        22-358   23-126   26-96    26-151   26-160   26-167   26-171   26-173   26-182   26-186   26-188   26-197   26-199   26-201
        26-211   26-215   26-217   28-124   29-23    29-25    29-27    29-37    29-39    29-41    30-68    30-106   30-144   30-153
        30-157   30-159   30-169   30-173   30-174   30-185   30-187   30-189   30-201   30-203   30-205   30-216   30-220   30-222
        31-23    32-161   32-194   32-204   32-210   32-212   32-214   32-226   32-230   32-232   32-242   32-246   32-248   32-258
        32-260   32-262   33-34    33-40    33-44    33-46    33-56    33-60    33-61*   33-62    33-72    33-76    33-78    56-13
RMDSO    7-0#
RMDT     4-691#  10-92    10-94    41-26
RMDTI    7-0#    56-5     56-17
RMDTO    7-0#
RMEC1    4-698#
RMEC1I   7-0#    56-14
RMEC10   7-0#
RMEC2    4-699#  16-15
RMEC2I   7-0#    13-186   16-14    28-92    31-94    56-15
RMEC20   7-0#
RMER1    4-687#  13-126   13-230   13-494   13-595   13-748   13-<63   13-B15   13-F41   21-143
```

CROSS REFERENCE TABLE (CREF V04.00)

```
RMER1I    7-0#    13-163  13-300  13-302  13-307  13-309  13-:31  13-;64  13-<30  13-=20  13-=23  13-=49  13-=52  13-=82
         21-135   21-150  21-152  22-80   22-156  22-183  22-226  22-241  22-284  22-361  22-387  22-402  22-417  22-498
         22-517   22-520  23-123  23-139  23-154  23-166  24-24   26-30   26-37   26-71   26-145  26-148  26-169  28-69
         29-51    29-55   29-57   29-58   29-69   29-71   29-72   29-83   29-85   29-86   29-97   29-99   29-100  29-111
         29-113   29-114  30-24   30-29   30-33   30-35   30-45   30-47   30-49   30-61   30-63   30-65   30-78   30-80
         30-82    30-155  30-232  30-236  30-238  30-240  30-250  30-252  30-254  30-264  30-268  30-270  31-41   32-50
         32-56    32-58   32-112  32-187  32-189  32-191  32-215  33-42   56-13
RMER10    7-0#    13-118* 13-223* 13-494* 13-589* 13-748* 13-<63* 13-B08* 13-F34* 21-139
RMER2     4-697#  13-56   13-65   13-127  13-231  13-594
RMER2I    7-0#    13-198  13-294  13-296  13-321  13-323  13-378  13-380  13-384  13-386  13-418  13-420  13-424  13-426
         13-440   13-442  13-529  13-:98  13-A92  13-A95  13-F18  13-F21  15-55   21-137  22-82   22-200  22-256  22-432
         22-460   24-27   26-32   26-90   26-93   26-104  26-132  26-135  26-184  26-213  28-114  30-31   30-92   30-99
         30-101   30-103  30-116  30-118  30-120  30-130  30-132  30-134  30-171  30-218  30-266  31-103  32-52   32-130
         32-134   32-154  32-156  32-158  32-171  32-173  32-175  32-228  32-244  33-58   33-74   33-88   33-94   33-96
         33-98    33-108  33-110  33-112  56-13
RMER20    7-0#    13-119* 13-224* 13-590* 13-591*
RMHR      4-695#
RMHRI     7-0#
RMHRO     7-0#
RMLA      4-689#  13-C14  13-C15  13-C24
RMLAI     7-0#    56-16
RMLAO     7-0#
RMMR1     4-690#  13-128  13-232  13-350  13-391  13-452  13-454  13-518  13-520  13-:00  13-:02  13-:66  13-:68  13-;33
         13-;35   13-;99  13-<01  13-=02  13-=04  13-=11  13-A74  13-A81  13-E95  13-F03
RMMR1I    7-0#    13-178  13-354  13-357  13-394  13-397  28-79   31-69   56-17
RMMR10    7-0#    13-120* 13-225* 13-350* 13-391* 13-452* 13-454* 13-518* 13-520* 13-:00* 13-:02* 13-:66* 13-:68* 13-;33*
         13-;35*  13-;99* 13-<01* 13-=02* 13-=04* 13-=11* 13-A74* 13-A77* 13-E95* 13-E97*
RMMR2     4-696#
RMMR2I    7-0#    13-190  28-101  31-81   56-17
RMMR20    7-0#
RMOF      4-693#  13-922  13-:18  13-:85  13-;51  13-<17  13-<75  13-@45  13-@94  13-A39  13-B88  13-C82  13-D25  13-D75
         13-E22   13-E60  13-F06  13-F80  18-51
RMOFI     7-0#    13-936  22-377  22-448  22-450  22-515  23-95   56-14   56-16
RMOFO     7-0#    13-927* 13-:14* 13-:81* 13-;47* 13-<13* 13-<71* 13-@41* 13-@77  13-@79* 13-@92* 13-A35* 13-B81* 13-C35
         13-C55   13-C57* 13-C60  13-C62* 13-C77* 13-D19* 13-D69* 13-E03  13-E05* 13-E18* 13-E58* 13-F00* 13-F75* 23-71
         23-98    26-59   26-63   32-100  32-104  56-11
RMR       4-573#  13-=20  13-=22  29-51   29-83   29-87   30-232  30-250  30-253
RMSN      4-692#
RMSNI     7-0#    56-17
RMSNO     7-0#
RMWC      4-759#  13-10*  13-11   13-:17  13-:84  13-;50  13-<16  13-<74
RMWCI     7-0#    23-10   23-23   23-44   23-149  56-14
RMWCO     7-0#    13-:11* 13-:78* 13-;44* 13-<10* 13-<68* 23-24   23-45
RQA       4-654#  13-191  28-102  31-82
RQB       4-655#  13-191  28-102  31-82
RTC       4-502#  13-888
SA1       4-538#
SA16      4-534#
SA2       4-537#
SA4       4-536#
SA8       4-535#
SADMSK    4-542#  23-68
SAVREG   41-13    44-1#
SC        4-710#  13-142  21-90   21-107  21-123  28-30
SC0       4-591#
SC1       4-590#
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
SW07      4-479     4-479#
SW08      4-479     4-479#
SW09      4-479     4-479#
SW1       4-479#
SW10      4-479#
SW11      4-479#
SW12      4-479#
SW13      4-479#    41-14
SW14      4-479#
SW15      4-479#
SW2       4-479#
SW3       4-479#
SW4       4-479#
SW5       4-479#
SW6       4-479#
SW7       4-479#
SW8       4-479#
SW9       4-479#
SWR       6-0#      10-24     10-24     10-24*    10-24*    10-24*    10-29     39-1      39-1      39-1      39-1      39-1      39-1      39-1*
         39-1*      39-1*     39-1*     40-1      40-1      40-1      40-1      40-1      41-14     42-1      42-1      42-1*     45-1      45-1*
SWREG     5-1#      10-24     10-29     42-1      42-1      42-1
SYSTAT   10-76      47-18#
TA1       4-531#    12-55
TA2       4-530#
TA4       4-529#    12-55
TA8       4-528#    12-55
TADMSK    4-541#    23-66
TAG       4-656#
TAGADR    5-9#      6-0
TAP       4-635#
TBITVE    4-479#
TIMOUT   13-335     13-458    13-479    13-503    13-524    13-550    13-572    13-601    13-637    13-795    13-856    13-890    13-931    13-:24
         13-:91     13-;57    13-<23    13-<83    13->26    13->56    13->87    13-?22    13-?54    13-?83    13-@17    13-@60    13-A09    13-A52
         13-B28     13-B56    13-B64    13-C06    13-C87    13-C97    13-D33    13-D45    13-D85    13-E31    13-E71    13-F12    13-F56    13-F86
         13-F93     15-136    15-194    20-9#
TKVEC     4-479#    42-1*     42-1*
TPVEC     4-479#
TRAPVE    4-479#    10-24*    10-24*
TRE       4-711#    13-256    13-266    21-90     21-107    21-123    22-106    22-108    23-12
TRTVEC    4-479#
TST       4-657#    13-192    28-103    31-83
TST1     13-2#      39-1
TST10    13-346#    39-1
TST11    13-448#    39-1
TST12    13-473#    39-1
TST13    13-490#    39-1
TST14    13-514#    39-1
TST15    13-536#    39-1
TST16    13-561#    39-1
TST17    13-585#    39-1
TST2     13-33#     39-1
TST20    13-613#    39-1
TST21    13-631#    39-1
TST22    13-662#    39-1
TST23    13-698#    39-1
TST24    13-733#    39-1
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

	13-92	13-93	13-95	13-97	14-20	14-20	14-20	35-1	36-1	37-1	38-1	40-1	40-1	41-20
	41-21	41-31	41-32	41-38	41-43	41-45	41-53	41-96	41-102	41-106	41-118	41-124	41-126	41-144
	41-146	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1	42-1
	42-1	42-1	42-1	42-1	42-1	44-1#	45-1							
TYPOC	10-8	11-33	11-47	41-138	42-1	44-1#								
TYPON	44-1#													
TYPOS	10-46	10-57	10-81	12-16	12-60	13-40	13-49	13-94	41-22	41-39	41-44	41-46	44-1#	
U0	4-736#													
U1	4-736#													
U2	4-736#													
UNS	4-561#	13-164	13-300	13-303	13-307	13-494	13-748	13-<63	13-808	13-836	13-F34	13-F64	28-70	29-51
	29-55	29-59	30-232	30-264	30-269									
UNTMSK	4-740#	13-71	13-90	13-=36	21-48	21-50								
UNTOFF	10-109	47-24#												
UNTON	10-120	47-25#												
UPE	4-723#	13-271	22-270											
USE	4-679#													
VV	4-555#	13-317	13-430	13-433	15-122	22-196	26-96	26-161	26-162	26-211	26-216	29-23	29-26	30-106
	30-145	30-146	30-169	30-175	31-29	32-161	32-205	32-206	32-242	32-247	33-35	33-36	33-56	33-61
WC	4-625#	13-179	28-80	31-70										
WCD	4-512#													
WCE	4-722#	13-271	22-169	48-77	48-78									
WCEHI	4-747#													
WCELO	4-748#													
WCF	4-570#	4-577	22-240											
WCH	4-513#													
WD	4-516#													
WH	4-517#													
WLE	4-564#	4-577	13-<29	13-<31	22-221	22-225	22-238	22-253	22-268	48-81	48-82			
WRL	4-550#	13-314	13-372	13-375	13-412	13-415	13-416	22-223						
XSIZ	10-70#	11-88	11-118											
XXDP	7-0#	10-34*	10-37*	10-38	10-40*	10-45	10-56	10-113	10-115					
Y	47-37#													

CZRNDAO RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE M-1 SEQ 0318
CROSS REFERENCE TABLE (CREF V04.00)

```
$$CMRE   5-10#
$$CMTM   5-10#    6-0       6-0       6-0       6-0       6-0
$$ESCA   4-479#
$$NEWT   4-479#   13-2      13-33     13-105    13-210    13-252    13-277    13-328    13-346    13-448    13-473    13-490    13-514    13-536
        13-561    13-583    13-613    13-631    13-662    13-698    13-733    13-765    13-830    13-884    13-915    13-959    13-988    13-:54
        13-;21    13-;87    13-<53    13-<95    13-=27    13-=60    13->11    13->41    13->71    13-?05    13-?39    13-?67    13-@01    13-@35
        13-@84    13-A28    13-A70    13-B04    13-B41    13-B77    13-C70    13-D14    13-D65    13-E10    13-E51    13-E91    13-F30    13-F69
$$SET    44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1      44-1#
$$SETM   10-24     10-24#
$$SKIP   4-479#
.$ACT1   4-471#    5-5
.$APTB   4-471#    6-0       6-0#
.$APTH   4-471#    5-8
.$APTY   4-471#   46-1
.$CATC   4-467#    5-1
.$CMTA   4-468#    5-10
.$EOP    4-468#   14-20
.$ERRO   4-468#   40-1
.$POWE   4-470#   45-1
.$RDDE   4-469#
.$RDOC   4-469#   43-1
.$READ   4-469#   42-1
.$SAVE   4-470#   34-1
.$SCOP   4-468#   39-1
.$SIZE   4-470#
.$TRAP   4-470#   44-1
.$TYPB   4-469#   35-1
.$TYPD   4-469#   36-1
.$TYPE   4-468#   38-1
.$TYPO   4-469#   37-1
.EQUAT   4-467#    4-479
.HEADE   4-467#    4-475
.SETUP   4-467#    4-769
.SWRHI   4-467#    4-476
.SWRLO   4-467#    4-476#    4-477
CALCLR   4-179#   13-107    13-212    13-240    13-254    13-279    13-330    13-348    13-450    13-995    13-:61    13-;28    13-;94    13-=72
        13-@52    13-A01    13-A44    13-A72    13-B79    13-D81    13-E27    13-E67
CALSUB   4-193#   13-114    13-132    13-138    13-219    13-236    13-243    13-245    13-249    13-262    13-283    13-337    13-340    13-343
        13-353    13-393    13-439    13-460    13-463    13-466    13-470    13-475    13-481    13-483    13-485    13-487    13-492    13-499
        13-505    13-507    13-509    13-511    13-516    13-526    13-528    13-533    13-538    13-548    13-552    13-554    13-556    13-558
        13-563    13-570    13-572    13-574    13-576    13-578    13-580    13-585    13-598    13-603    13-606    13-608    13-610    13-615
        13-620    13-622    13-624    13-626    13-628    13-633    13-639    13-641    13-655    13-657    13-659    13-664    13-669    13-677
        13-681    13-683    13-691    13-693    13-695    13-700    13-705    13-718    13-720    13-722    13-730    13-735    13-740    13-752
        13-760    13-762    13-767    13-791    13-806    13-808    13-816    13-818    13-820    13-822    13-824    13-832    13-877    13-886
        13-892    13-894    13-908    13-910    13-912    13-917    13-929    13-933    13-935    13-952    13-954    13-956    13-961    13-966
        13-968    13-979    13-990    13-:23    13-:26    13-:28    13-:56    13-:90    13-:93    13-:95    13-:23    13-:56    13-:59    13-:61
        13-:89    13-<22    13-<25    13-<27    13-<55    13-<58    13-<61    13-<80    13-<83    13-<85    13-<87    13-<97    13-=00    13-=08
        13-=17    13-=29    13-=33    13-=45    13-=47    13-=62    13-=76    13-=78    13-=80    13->14    13->24    13->28    13->30    13->32
        13->34    13->36    13->44    13->54    13->58    13->60    13->62    13->64    13->66    13->75    13->84    13->89    13->91    13->93
        13->95    13->97    13-?08    13-?20    13-?24    13-?26    13-?28    13-?30    13-?32    13-?42    13-?52    13-?56    13-?58    13-?60
        13-?62    13-?64    13-?70    13-?81    13-?85    13-?87    13-?89    13-?91    13-?93    13-@04    13-@15    13-@19    13-@21    13-@23
        13-@25    13-@27    13-@38    13-@54    13-@56    13-@58    13-@62    13-@64    13-@66    13-@69    13-@71    13-@86    13-A03    13-A05
        13-A07    13-A11    13-A13    13-A15    13-A18    13-A20    13-A30    13-A46    13-A48    13-A50    13-A54    13-A56    13-A58    13-A60
        13-A62    13-A87    13-A89    13-A91    13-A99    13-B01    13-B06    13-B19    13-B21    13-B30    13-B32    13-B34    13-B36    13-B38
        13-B44    13-B54    13-B61    13-B64    13-B66    13-B68    13-B70    13-B72    13-B74    13-B93    13-C04    13-C08    13-C10    13-C72
        13-C86    13-C89    13-C91    13-C93    13-C95    13-C99    13-D01    13-D03    13-D05    13-D07    13-D16    13-D31    13-D35    13-D37
```

CZRNDAO RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE M-2 SEQ 0319
CROSS REFERENCE TABLE (CREF V04.00)

```
                13-D39  13-D43  13-D47  13-D49  13-D51  13-D53  13-D55  13-D67  13-D83  13-D87  13-D89  13-D91  13-D93  13-D95
                13-D97  13-E12  13-E29  13-E33  13-E35  13-E37  13-E39  13-E41  13-E43  13-E53  13-E69  13-E73  13-E75  13-E77
                13-E79  13-E81  13-E83  13-E93  13-F10  13-F14  13-F16  13-F25  13-F27  13-F32  13-F45  13-F47  13-F58  13-F60
                13-F62  13-F64  13-F66  13-F71  13-F83  13-F91  13-F95  13-F97  13-F99  13-G01  13-G03
COMMEN           4-479#
ENDCOM           4-479#
ERROR            4-479#  13-25   13-59   13-68   13-75   13-82   13-87   13-107  13-114  13-132  13-138  13-146  13-151  13-160
                13-166  13-176  13-183  13-188  13-195  13-201  13-207  13-212  13-219  13-236  13-240  13-243  13-245  13-249
                13-254  13-256  13-262  13-268  13-273  13-279  13-283  13-291  13-298  13-304  13-311  13-319  13-325  13-330
                13-332  13-337  13-340  13-343  13-348  13-350  13-353  13-358  13-365  13-371  13-377  13-383  13-389  13-391
                13-393  13-398  13-405  13-411  13-417  13-423  13-429  13-435  13-437  13-439  13-445  13-450  13-452  13-454
                13-456  13-460  13-463  13-466  13-470  13-475  13-477  13-481  13-483  13-485  13-487  13-492  13-494  13-496
                13-499  13-502  13-505  13-507  13-509  13-511  13-516  13-518  13-520  13-522  13-526  13-528  13-531  13-533
                13-538  13-548  13-552  13-554  13-556  13-558  13-563  13-565  13-567  13-574  13-576  13-578  13-580  13-585
                13-598  13-603  13-606  13-608  13-610  13-615  13-617  13-620  13-622  13-624  13-626  13-628  13-633  13-635
                13-639  13-641  13-647  13-653  13-655  13-657  13-659  13-664  13-666  13-669  13-675  13-677  13-679  13-681
                13-683  13-689  13-691  13-693  13-695  13-700  13-702  13-705  13-711  13-718  13-720  13-722  13-728  13-730
                13-735  13-737  13-740  13-746  13-748  13-750  13-752  13-758  13-760  13-762  13-767  13-791  13-793  13-806
                13-808  13-809  13-816  13-818  13-820  13-822  13-824  13-832  13-854  13-863  13-877  13-878  13-886  13-888
                13-892  13-894  13-900  13-906  13-908  13-910  13-912  13-917  13-929  13-933  13-935  13-940  13-945  13-950
                13-952  13-954  13-956  13-961  13-963  13-966  13-968  13-974  13-977  13-979  13-985  13-990  13-995  13-:00
                13-:02  13-:06  13-:23  13-:26  13-:28  13-:35  13-:46  13-:56  13-:61  13-:66  13-:68  13-:90  13-:93  13-:95
                13-;02  13-;13  13-;23  13-;28  13-;33  13-;35  13-;39  13-;56  13-;59  13-;61  13-;68  13-;79  13-;89  13-;94
                13-;99  13-<01  13-<05  13-<22  13-<25  13-<27  13-<34  13-<45  13-<55  13-<61  13-<63  13-<80  13-<85  13-<87
                13-<97  13-=02  13-=04  13-=06  13-=08  13-=11  13-=13  13-=15  13-=17  13-=24  13-=29  13-=33  13-=45  13-=47
                13-=54  13-=62  13-=72  13-=76  13-=78  13-=80  13-=87  13->14  13->24  13->28  13->30  13->32  13->34  13->36
                13->44  13->54  13->58  13->60  13->62  13->64  13->66  13->75  13->84  13->89  13->91  13->93  13->95  13->97
                13-?08  13-?20  13-?24  13-?26  13-?28  13-?30  13-?32  13-?42  13-?52  13-?56  13-?58  13-?60  13-?62  13-?64
                13-?70  13-?81  13-?85  13-?87  13-?89  13-?91  13-?93  13-a04  13-a15  13-a19  13-a21  13-a23  13-a25  13-a27
                13-a38  13-a52  13-a54  13-a56  13-a58  13-a62  13-a64  13-a66  13-a69  13-a71  13-a86  13-A01  13-A03  13-A05
                13-A07  13-A11  13-A13  13-A15  13-A18  13-A20  13-A30  13-A44  13-A46  13-A48  13-A50  13-A54  13-A56  13-A58
                13-A60  13-A62  13-A72  13-A74  13-A87  13-A89  13-A91  13-A97  13-A99  13-B01  13-B06  13-B19  13-B21  13-B27
                13-B30  13-B32  13-B34  13-B36  13-B38  13-B44  13-B54  13-B61  13-B66  13-B68  13-B70  13-B72  13-B74  13-B79
                13-B93  13-C04  13-C08  13-C10  13-C28  13-C50  13-C72  13-C86  13-C89  13-C91  13-C93  13-C95  13-C99  13-D01
                13-D03  13-D05  13-D07  13-D16  13-D31  13-D35  13-D37  13-D39  13-D43  13-D47  13-D49  13-D51  13-D53  13-D55
                13-D67  13-D81  13-D83  13-D87  13-D89  13-D91  13-D93  13-D95  13-D97  13-E12  13-E27  13-E29  13-E33  13-E35
                13-E37  13-E39  13-E41  13-E43  13-E53  13-E67  13-E69  13-E73  13-E75  13-E77  13-E79  13-E81  13-E83  13-E93
                13-E95  13-F10  13-F14  13-F16  13-F23  13-F25  13-F27  13-F32  13-F45  13-F47  13-F54  13-F58  13-F60  13-F62
                13-F64  13-F66  13-F71  13-F83  13-F91  13-F95  13-F97  13-F99  13-G01  13-G03  39-1
ESCAPE           4-479#
GETPRI           4-479#
GETREG           4-156#
GETSWR           4-479#  10-29   10-29#
MSG              4-8#
MULT             4-479#
NEWTST           4-479#  13-2    13-33   13-105  13-210  13-252  13-277  13-328  13-346  13-448  13-473  13-490  13-514  13-536
                13-561  13-583  13-613  13-631  13-662  13-698  13-733  13-765  13-830  13-884  13-915  13-959  13-988  13-:54
                13-:21  13-;87  13-<53  13-<95  13-=27  13-=60  13->11  13->41  13->71  13-?05  13-?39  13-?67  13-a01  13-a35
                13-a84  13-A28  13-A70  13-B04  13-B41  13-B77  13-C70  13-D14  13-D65  13-E10  13-E51  13-E91  13-F30  13-F69
NWTST            4-36#  13-2    13-33   13-105  13-210  13-252  13-277  13-328  13-346  13-448  13-473  13-490  13-514  13-536
                13-561  13-583  13-613  13-631  13-662  13-698  13-733  13-765  13-830  13-884  13-915  13-959  13-988  13-:54
                13-:21  13-;87  13-<53  13-<95  13-=27  13-=60  13->11  13->41  13->71  13-?05  13-?39  13-?67  13-a01  13-a35
                13-a84  13-A28  13-A70  13-B04  13-B41  13-B77  13-C70  13-D14  13-D65  13-E10  13-E51  13-E91  13-F30  13-F69
POP              4-479#  13-17   13-20   13-24   13-;08  13->09  13-C66  13-C67  16-25   16-26   16-27   17-70   18-63   19-18
                20-31   20-44   21-145  21-149  22-131  22-447  25-51   27-25   28-58   31-58   31-59   34-1    36-1    43-1
                45-1    45-1    46-1    46-1
```

CZRNDAO RM80 FCTNL PT1 MACRO V04.00 28-JAN-82 16:24:51 PAGE M-3 SEQ 0320
CROSS REFERENCE TABLE (CREF V04.00)