

# RH11-RS03/4

RH11-RS03-RS03/LA-RS04  
MD-11-DZRSB-F  
BASIC FUNCTION

EP-DZRSB-F-DL

COPYRIGHT © 73-77

FICHE 1 OF 1

DEC 1977

**digital**

MADE IN USA

This microfiche card contains a grid of frames. The first column contains text, likely labels for the diagrams. The subsequent columns contain technical diagrams, possibly circuit board layouts or component schematics, showing various components and their interconnections. The diagrams are arranged in a regular grid pattern across the card.

B01

EOFDZRFZCSEG

00010000

771114

POP10 411

ONMORIDZRSBFSEG

00010000

771114

DZRSB P11 2-SEP-- 15:12

MAINDEC-11-DZRSB-E  
DZRSB P11 2-SEP-- 15:12

.REM 2

IDENTIFICATION  
-----

PRODUCT CODE: MAINDEC-11-DZRSB-F-D  
PRODUCT NAME: RH11-RS03-RS03/LA-RS04 BASIC FUNCTION  
DIAGNOSTIC  
DATE CREATED: JULY 1977  
MAINTAINER: DIAGNOSTIC GROUP

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

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

COPYRIGHT (C) 1973, 1974, 1975, 1976, 1977 BY DIGITAL EQUIPMENT CORPORATION

MAINDEC-11-DZRSB-F RH11-R503 LA-R504 BASIC FUNCTION DIAGNOSTIC PAGE 2  
TABLE OF CONTENTSCONTENTS  
-----

1.	ABSTRACT
2.	REQUIREMENTS
2.1	EQUIPMENT
2.3	PRELIMINARY PROGRAMS
3.	LOADING PROCEDURE
4.	STARTING PROCEDURE
4.1	CONTROL SWITCH SETTINGS
4.2	STARTING ADDRESS
4.3	PROGRAM AND/OR OPERATING PROCEDURE
5	OPERATIONAL SWITCH SETTINGS
5.2	SUBROUTINE ABSTRACT
6.	ERRORS
7.	RESTRICTIONS
8.	MISCELLANEOUS
8.1	EXECUTION TIME
8.2	STACK POINTER
9.	WRITE LOCK TEST
10.	TEST DESCRIPTION

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 3  
DESCRIPTION

## 1. ABSTRACT

THIS DIAGNOSTIC WAS DESIGNED TO TEST RS03,RS03/LA AND RS04 DRIVES.

THIS IS A BASIC FUNCTION DIAGNOSTIC WHICH IS USED TO VERIFY THAT THE (RH11) CONTROLLER AND THE (RS03,RS03/LA OR RS04) DISKS ARE OPERATING CORRECTLY. THIS IS NOT A RELIABILITY DIAGNOSTIC AND THEREFORE SHOULD NOT BE USED AS ONE. THIS PROGRAM CAN TEST UP TO 8 DRIVES. THE DRIVES CAN BE INTERMIXED AND IN ANY ORDER.

IF THE OPERATOR WOULD LIKE TO CHECK THE DISK REGISTERS PRIOR TO ENTERING THIS DIAGNOSTIC, THERE ARE SOME ROUTINES IN THE BACK OF THE DIAGNOSTIC WHICH CAN BE USED. THESE ROUTINES WILL ALLOW THE OPERATOR TO LOAD THE REGISTERS THROUGH THE SWITCHES. PLEASE REFERENCE THE STARTING ADDRESSES THAT WILL TEST THE REGISTERS YOU DESIRE.

## 2. REQUIREMENTS

### 2.1 EQUIPMENT

PDPI1 STANDARD COMPUTER WITH A MINIMUM OK BK OF MEMORY. AND AN RH11 CONTROLLER WITH A RS03, RS03/LA OR RS04 DISK.

### 2.3 PRELIMINARY PROGRAMS

NONE

## 3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

## 4. STARTING PROCEDURE

### 4.1 CONTROL SWITCH SETTINGS

SEE 5.1.1 (ALL DOWN FOR WORST CASE TESTING)

### 4.2 STARTING ADDRESS

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 4  
 DESCRIPTION

## 4.3 PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY USING ABS LOADER.

## 1. STARTING ADDRESS 200.

- A. SET SWITCHES (SEE SEC 5.1.1) ALL DOWN FOR WORST CASE. IF SWITCHLESS CPU, SIMPLY
- B. PRESS START.
- C. THE PROGRAM WILL LOOP AND BELL WILL RING ONCE EVERY PASS
- D. THE DISPLAY ON THE 11/45 WILL SHOW THE ITERATION COUNT IN THE LEFT BYTE AND TEST NUMBER IN THE RIGHT. TO USE, SET THE DATA DISPLAY SWITCH TO THE DISPLAY POSITION.
- E. THE PROGRAM WILL TEST ALL RS03, RS03/LA AND RS04 DISKS.

## 2. STARTING ADDRESSES FOR TESTING THE RH11-RS03/LA/04 REGISTERS USING THE SWITCH REGISTER. ON SWITCH-LESS MACHINES THESE ROUTINES ARE USEFULL FOR SCOPING. SIMPLY STRIKE 'G' ANYTIME AFTER PRESSING START TO ENTER OR CHANGE VALUE DESIRED.

A.	250	WORD COUNT REGISTER TEST
B.	254	BUS ADDRESS REG. TEST
C.	260	DISK ADDRESS REG. TEST
D.	264	DAIVE STATUS REG. TEST
E.	270	ERROR REG. TEST
F.	274	LOOK AHEAD REG. TEST
G.	300	RSCS2 REG. TEST
H.	304	ATTENTION SUMMARY REG. TEST
I.	310	MAINTENANCE REG. TES
J.	314	RSCS1 REG TEST

## 5. OPERATIONAL SWITCH SETTINGS

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC.176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(I.E.) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

- 1. CR) IF NO CHANGES ARE TO BE MADE.
- 2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE; LAST DIGIT FOLLOWED BY 'CR'.
- 3. 'G' TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED

GO1

MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 46-1 SEQ 0005  
DZRSB.F11 28-SEP-77 15:12

KEYING IN SWREG VALUE.

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 5  
 DESCRIPTION

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING IG (CNTRL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

SWITCH SETTINGS ARE:

- SW<15> = 1 ..... HALT ON ERROR
- SW<14> = 1 ..... LOOP ON TEST
- SW<13> = 1 ..... INHIBIT TYPEOUTS
- SW<12> = 1 ..... INHIBIT OBUF SV FROM CHANGING WHEN LOOKING FOR MEMORY ON -B- PORT
- SW<11> = 1 ..... INHIBIT ITERATIONS OF SUBTEST
- SW<10> = 1 ..... BELL ON ERROR
- SW<09> = 1 ..... BELL ON PASS COMPLETE
- SW<08> = 1 ..... LOOP ON ERROR
- SW<07> = 1 ..... LOOP ON TEST IN SW<7:0>

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED IN LOCATION "LAD". IF A SCOPE LOOP IS REQUESTED THE CURRENT SUBTEST WILL BE LOOPED UPON. SW<11> ON A 1 INHIBITS ITERATION OF SUBTESTS. THE CONTENTS OF LAD MAY BE USED TO DETERMINE THE LAST SUBTEST SUCCESSFULLY COMPLETED.

5.2.2 HLT

THIS ROUTINE PRINTS OUT AN ERROR MESSAGE (SEE 6.1). TO INHIBIT TYPEOUTS, PUT SW<13> ON A 1.

5.2.3 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

6. ERRORS

MAINDEC-11-DZRSB-F RH11-RS03-LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 6  
 DESCRIPTION

### 6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

```
ADR   CS1 = ----- CS2 = ----- ER = -----
GOOD   = ----- BAD = -----
```

WHERE:

```
CS1, CS2, ER ETC.   = RS11 DISK REGISTERS.
GOOD                = EXPECTED DATA.
BAD                 = DATA RECEIVED.
```

TO FIND THE FAILING TEST, LOOK AT THE LISTING ABOVE THE ADDRESS TYPED.

### 6.2 ERROR RECOVERY

RESTART AT 200

### 7. RESTRICTIONS

NONE

### 8. MISCELLANEOUS

#### 8.1 EXECUTION TIME

A BELL WILL RING WITHIN 1 MINUTE WITH ALL SWITCHES DOWN.

#### 8.2 STACK POINTER

STACK IS INITIALLY SET TO 500

### 9. WRITE LOCK TEST

THE WRITE LOCK TEST REQUIRES OPERATOR INTERVENTION. THE STARTING ADDRESS FOR THIS TEST IS 220. THE PROGRAM WILL TELL THE OPERATOR WHICH SWITCHES HAVE TO BE SET.

J01

MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 48-1 SEG 0008  
DZRSBF.P11 27-SEP-77 15:12

10 TEST DESCRIPTION

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 7  
DESCRIPTION

1. TEST RSCS2  
CLEAR ALL READ/WRITE BITS AND CHECK. SET ALL R/W BITS AND CHECK. NOW CLEAR AND RECHECK.
2. TEST FOR ONLINE DRIVES  
SET ERROR BITS IN RSER. THIS CAUSES ATTENTION SUMMARY BITS TO SET IN RSAS. DO FOR ALL DRIVES. RSAS HAS NOT YET BEEN TESTED. SO IN THE CASE OF NO BITS IN RSAS SETTING, DRIVE 0 IS TESTED.
3. RESET TEST FOR REGISTERS  
SET ALL R/W BITS IN RSCS1, RSCS2, RSBA, RSDA, RSER, RSWC, RSDB, AND RSMR. DO A RESET AND TEST ALL R/W BITS TO BE CLEARED.
4. SET AND CLEAR ALL REGISTERS  
SET ALL R/W BITS IN RSCS1, RSCS2, RSBA, RSDA, RSER, RSWC, RSDB AND RSMR AND TEST. SET ALTERNATE BITS AND CHECK TO MAKE SURE BITS ARE NOT TIED TOGETHER. NOW SET ALL BITS AND CLEAR THEM TO MAKE SURE ALL CAN BE CLEARED ONCE SET.
5. RANDOM NUMBER TEST FOR RSWC AND RSDA  
THIS TEST GENERATES RANDOM NUMBERS AND LOADS THEM INTO RSWC, RSDA AND RSBA.
6. TEST "CLEAR BIT" IN RSCS2  
SET ALL R/W BITS IN RSCS1, RSCS2, RSBA, RSDA, RSER, RSWC, RSDB, AND RSMR. SET CLEAR BIT IN RSCS2. NOW TEST ALL R/W BITS FOR 0 IN ALL THE ABOVE REGISTERS.
7. TEST DLT AND TRE BITS  
DO A READ FROM THE SILO. THIS SHOULD CAUSE A DLT AND A TRE ERROR BECAUSE THE SILO IS EMPTY.
8. CLEAR DLT AND TRE  
CLEAR BY SETTING TRE IN RSCS1 AND TEST.
9. LOAD RSDB WITH ALL ONES AND ALL ZEROS  
LOAD RSDB WITH A WORD OF ZEROS AND A WORD OF ONES. WAIT FOR "OR" TO SET AND THEN CHECK OUTPUT OF SILO. IF OR DID NOT SET ERROR MESSAGE APPEARS.
10. TEST FOR 66 LOCATIONS IN SILO

L01

MAINDEC-11-DZRSB-F      RM11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 49-1      SEQ 0010  
DZRSBF.P11      27-SEP-77 15:12

THIS IS DONE BY PUTTING A BINARY COUNT IN EVERY LOCATION AND

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 8  
 DESCRIPTION

CHECKING THE OUTPUT FOR 66 WORDS.

11. TEST DLT ERROR

THIS IS DONE BY LOADING THE SILO WITH 67 WORDS WITHOUT READING ANY OUT. THIS SHOULD CAUSE DLT TO SET.

12. FLOAT A "1" AND A "0" THROUGH THE SILO

LOAD THE SILO WITH A WORD OF ZEROS AND FLOAT A "1" THROUGH THE WORD. THEN LOAD THE SILO WITH A WORD OF ALL ONES AND FLOAT A "0" THROUGH THE WORD. CHECK THE OUTPUT OF THE SILO FOR THE CORRECT ANSWER.

13. TEST NO-OP FUNCTION

THE NO-OP FUNCTION IS TESTED WITH AND WITHOUT ERROR BITS SET. ALL THE REGISTERS ARE CHECKED AFTER BOTH CASES.

14. TEST DRIVE CLEAR FUNCTION

FIRST SET ALL R/W BITS IN RSDA, RSWC, RGER, AND RSMR. DO A DRIVE CLEAR FUNCTION. NOW TEST ALL REGISTERS FOR CORRECT DATA.

15. EXECUTE A ONE WORD WRITE FUNCTION

SET RSWC TO -1. MOV -1 INTO OUTBUF. LOAD RSBA WITH OUTBUF. DO A WRITE TEST RDY BIT FOR 0 THEN WAIT FOR IT TO SET. TIME OUT TO ERROR IF RDY BIT DOESN'T SET AND CHECK FOR ERROR CONDITIONS. TEST RSDA FOR CORRECT ADDRESS. TEST WORD COUNT FOR 0. THIS IS TESTED ON -A- AND -B- PORT.

16. EXECUTE A ONE WORD WRITE CHECK

SET UP RSDA, RSBA, RSWC AND OUTBUF AS IN THE WRITE FUNCTION TEST. DO A WRITE CHECK FUNCTION. TEST RDY AS DONE IN THE WRITE TEST. CHECK FOR WRITE CHECK ERROR. THEN TEST RSDA, RSWC AND RSBA FOR CORRECT DATA. THIS IS TESTED ON -A- AND -B- PORT.

17. TEST READ FUNCTION

SETUP RSDA, RSBA, RSWC AND OUTBUF AS IN THE WRITE FUNCTION DO A READ FUNCTION. TEST RDY BIT AS DONE IN THE WRITE FUNCTION. TEST FOR ERRORS. ALSO TEST RSDA, RSWC AND RSBA FOR CORRECT DATA. THIS IS TESTED ON -A- AND -B- PORT.

18. TEST BLOCK SEARCH FUNCTION, PIP AND DRY BITS AND ADDR. CONF. BIT

DO A BLOCK SEARCH FOR SECTOR 32. LOOP ON ADDR. CONF. BIT IN

NO1

MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 50-1 SEQ 0012  
DZRSBF.P11 27-SEP-77 15:12

RSMR. IF IT DOESN'T SET, TIMEOUT. WHEN YOU GET THERE DO A  
BLOCK SEARCH FOR SECTOR 0. NOW WE KNOW THAT WE HAVE TIME TO

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 9  
 DESCRIPTION

TEST FOR DRY AND PIP BITS BEFORE FINDING SECTOR 0. FOR PIP SHOULD SET AND DRY SHOULD CLEAR BEFORE FINDING SECTOR 0. ONCE SECTOR 0 IS FOUND PIP SHOULD CLEAR AND DRY SHOULD SET. IF DRY DOES NOT SET A TIME OUT ERROR WILL OCCUR INDICATING SECTOR 0 WAS NOT FOUND. SC IN RSCS1 SHOULD ALSO SET. RSBA AND RSWC SHOULD NOT MOVE, THIS IS ALSO TESTED.

19. ILLEGAL FUNCTION CODE TEST

IN THIS TEST RSBA, RSWC AND RSDA ARE SET UP AS IF TO DO A LEGAL FUNCTION. AN ILLEGAL FUNCTION IS THEN EXECUTED. THE PROGRAM TEST FOR ILF AND ERR BITS TO SET. RSBA, RSWC AND RSDA ARE ALSO TESTED FOR CORRECT DATA. THIS IS DONE FOR ALL THE ILLEGAL FUNCTIONS.

FOR AN AID IN TROUBLE SHOOTING THE ILLEGAL FUNCTION CODE CAN BE LOADED INTO LOCATION ILLTAB OR ILFTB2. DEPENDING ON WHICH ILLEGAL FUNCTION TEST YOU WISH TO LOOP ON. IN THE NEXT LOCATION, FOLLOWING THE ILLEGAL FUNCTION, A 0 MUST BE LOADED. NOW BY SETTING SWITCH 14 (LOOP ON TEST), YOU WILL LOOP ON THE ILLEGAL FUNCTION.

20. TEST PAR IN RSER

SET PAR IN RSER AND CHECK. ALSO TEST ERR IN RSDS TO SET BECAUSE OF THE PAR SETTING.

21. TEST DPR AND MOL IN RSDS

BOTH THESE BITS SHOULD BE SET IN RSDS IF THE DRIVE IS ON LINE AND UP TO SPEED.

22. LOOK AHEAD TEST

FIRST CHECK TO SEE IF SECTOR FRACTION BITS ARE MOVING. NOW SET RSDA TO 0 AND INCREMENT IT EVERY TIME THE ADDR.CONF BIT SETS. IF THE ADDR.CONF BIT DOES NOT SET IN A CERTAIN LENGTH OF TIME, A TIME OUT ERROR OCCURS.

23. PARITY TEST

THIS WILL TEST THE PARITY LOGIC ONLY IF THERE IS PARITY MEMORY ON THE SYSTEM IN LESS THAN 28K. IT WILL WRITE BAD PARITY IN A MEMORY LOCATION THEN TRY TO DO A WRITE TO THE DRIVE FROM THAT LOCATION. THIS SHOULD CAUSE A PARITY ERROR.

24. TEST WRITE CHECK ERROR

IN THIS TEST THE PROGRAM WRITES A -1 ON TO THE DISK. A 0 IS NOW FLOATED THROUGH THE WORD IN THE BUS ADDRESS LOCATION, AND A WRITE CHECK FUNCTION IS DONE. THE WCE BIT IN RSCS2 SHOULD SET AND SHOULD CAUSE THE TRE BIT IN RSCS1 TO SET. THESE BITS

C02

MAINDEC-11-DZRSB-F RM11-R503LA-R503-R504 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 51-1 SEG 0014  
DZRSB.P11 27-SEP-77 15:12

ARE THEN CLEARED. A WORD OF 0 IS NOW WRITTEN ON THE DISK AND  
A 1 IS FLOATED THROUGH THE WORD IN THE BUS ADDRESS AND THE

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 10  
DESCRIPTION

WRITE CHECK FUNCTION TEST IS REPEATED.

## 25. TEST PROGRAM ERROR BIT IN RSCS2

HERE THE PROGRAM ATTEMPTS TO INITIATE A DATA TRANSFER OPERATION WHILE THE CONTROL IS CURRENTLY PERFORMING ONE. THIS SHOULD CAUSE PGE TO SET WHICH SHOULD CAUSE TRE TO SET. THESE BITS ARE THEN CLEARED. RSWC IS ALSO TESTED FOR IT SHOULD NOT BE 0 FOR THE CURRENT OPERATION SHOULD HAVE BEEN ABORTED DUE TO THE PGE ERROR.

## 26. TEST RMR IN RSER

HERE A WRITE COMMAND IS GIVEN AND DURING ITS EXECUTION THE PROGRAM TRIES TO MODIFY THE RSDA REG. THIS SHOULD CAUSE THE RMR BIT TO SET WHICH CAUSES THE ERR BIT TO SET. THESE BITS ARE THEN CLEARED.

## 27. TEST DCK IN RSER

HERE A WRITE COMMAND IS GIVEN THEN DURING THIS FUNCTION A DRIVE CLEAR COMMAND IS GIVEN. THIS SHOULD CAUSE THE DCK BIT TO SET WHICH SHOULD CAUSE THE ERR BIT TO SET. THESE BITS ARE THEN CLEARED.

## 28. TEST DISK ADDRESS REGISTER

LOAD THE LAST DISK ADDRESS (7777) INTO RSDA. DO A ONE WORD WRITE AND CHECK THAT RSDA INCREMENTED TO 10000.

## 29. TEST IAE ERROR

DO A ONE WORD WRITE BUT FIRST SET RSDA TO AN INVALID ADDRESS SUCH AS 10000. THIS SHOULD CAUSE A IAE ERROR WHICH WILL CAUSE ERR, ATA AND SC BITS TO SET. THESE BITS ARE THEN CLEARED BY LOADING A 1 INTO ATA IN RSAS.

## 30. TEST FOR NON-EXISTENT DISK ERROR

FIRST FIND A DRIVE THAT IS NOT ON THE SYSTEM OR OFF LINE. NOW TRY TO DO A ONE WORD WRITE TO THAT DRIVE. NED IN RSCS2 SHOULD SET WHICH SHOULD CAUSE TRE TO SET. THESE BITS ARE THEN CLEARED BY MOVING A 1 INTO TRE.

## 31. TEST DAO IN RSER AND LBT IN RSDS

SET RSDA TO ITS LAST ADDRESS. NOW WRITE ONE SECTOR PLUS ONE WORD. DAO SHOULD SET AND LBT SHOULD SET. THESE SHOULD CAUSE ERR, ATA, TRE AND SC TO SET. THESE ARE CLEARED BY DOING A CLEAR.

## 32. TEST BAI IN RSCS2

E02

MAINDEC-11-DZRSB-F      RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 52-1      SEQ 0016  
DZRSBF.P11      27-SEP-77 15:12

SET BAI IN RSCS2. DO A ONE WORD WRITE AND CHECK RSBA TO SEE

MAINDEC-11-DZRSB-F RH11-RS03/LA-RS04 BASIC FUNCTION DIAGNOSTIC PAGE 11  
 DESCRIPTION

IF IT INCREMENTED.

33. TEST NON-EXISTENT MEMORY ERROR BIT IN RSCS2

SET BITS A16 AND A17 IN RSCS1 FOR AN 18 BIT ADDRESS. MOV 173000 INTO RSBA. MOV -1000 INTO RSAC AND DO A WRITE FUNCTION. THE NEM BIT SHOULD SET AND SHOULD CAUSE TRE TO SET. CLEAR THESE BITS BY LOADING A 1 INTO TRE.

34. TEST FOR ZERO'S IN A PARTIALLY FILLED SECTOR

FIRST WRITE A COMPLETE SECTOR WITH ALL ONES. THEN DO A ONE WORD WRITE. THE REMAINING 63 WORDS SHOULD BE WRITTEN AS ZERO'S. NOW DO A WRITE CHECK TO COMPARE FOR THESE ZERO'S.

35. PRIORITY INTERRUPT TEST

HERE THE PROGRAM ENABLES THE INTERRUPT AND DOES A ONE WORD WRITE FUNCTION. THE PROGRAM SHOULD NOT TRAP UNTIL THE PROCESSOR IS DROPPED TO PRIORITY 4.

36. DYNAMIC FUNCTION TEST

WHILE ONE DRIVE IS READING, THE UNIT # IN RSCS2 IS MODIFIED. IF THERE IS ANOTHER DRIVE ON THE SYSTEM, A DRIVE SEARCH IS PERFORMED ON IT. THIS IS ALL DONE WHILE THE FIRST DRIVE IS STILL READING.

TITLE MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC  
 :COPYRIGHT 1973, 1974, 1975, 1976, 1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.  
 :PROGRAM BY STANLEY KARACKIEWICZ

	SWITCH	USE
:		
:		
	SW15= 100000	:HALT ON ERROR
	SW14= 40000	:LOOP ON TEST
	SW13= 20000	:INHIBIT ERROR TYPEOUTS
	SW12= 10000	:INHIBIT OBUFSV FROM CHANGING FOR -B- PORT ONLY
	SW11= 4000	:INHIBIT ITERATIONS
	SW10= 2000	:0 - BELL ON PASS COMPLETE
		:1 - BELL ON ERROR
	SW9= 1000	:LOOP ON ERROR
	SW8= 400	:LOOP ON TEST IN SW(7:0)
		:TRAP CATCHER FROM 0 - 776
		:HOOKS FOR ACT 11
000046	SENDAD	
000052	BIT14	
000174	DISPREG:0	:SOFTWARE SWITCH REGISTER LOCATION

G02

```

000176 SWREG: 0
000200 . = 200
      JMP 200STRT
000220 . = 220
      JMP 200WRTLCK :WRITE LOCK TEST
000230 . = 230
000236 STRT: BIC 200BIT11 ONCEE :CLEAR TKSEL TEST FLAG
      JMP 200BEGIN :STARTING ADDRESS IS BEGIN

:STARTING ADDRESSES FOR SWITCH REGISTER TESTING OF RH-RS REGISTERS

000250 . = 250
000250 JMP 200SRSWC :WORD COUNT REG
000254 JMP 200SR5BA :BUS ADDR REG
000260 JMP 200SR5DA :DISK ADDR REG
000264 JMP 200SR5DS :DRIVE STATUS REG
000270 JMP 200SR5ER :ERROR REG
000274 JMP 200SR5LA :LOOK AHEAD REG
000300 JMP 200SRCS2 :CS2 REG
000304 JMP 200SRAS :ATTENTION SUMMARY REG
000310 JMP 200RSMR :MAINTENANCE REG
000314 JMP 200SRSCS1 :CSI REG

000320 . = 320
00032E BIS 200BIT11 ONCEE :SET TKSEL TEST FLAG
      JMP 200BEGIN :ENABLES OPERATOR TO DO A
      :WRITE OR READ TO A DESIRED ADDR
      :OPTIONAL WC, DA, AND UNIT #
  
```

H02

N#  
HLT= EMT  
PS= 177776  
PSW= PS  
BELL= 7  
R0= %0  
R1= %1  
R2= %2  
R3= %3  
R4= %4  
R5= %5  
SP= %6  
PC= %7  
BIT0= 1  
BIT1= 2  
BIT2= 4  
BIT3= 10  
BIT4= 20  
BIT5= 40  
BIT6= 100  
BIT7= 200  
BIT8= 400  
BIT9= 1000  
BIT10= 20000  
BIT11= 40000  
BIT12= 100000  
BIT13= 200000  
BIT14= 400000  
BIT15= 1000000

GOOD= %1  
BAD= %0

: INITIALIZE FOR NEWTST  
: SET HLT TO EMT FOR ERROR TYPEOUTS  
: PROCESSOR STATUS  
: PROCESSOR STATUS WORD  
: BELL  
: R0 - DEFINE REGISTERS  
: R1  
: R2  
: R3  
: R4  
: R5  
: R6 - STACK POINTER  
: R7 - PROGRAM COUNTER  
: BIT EQUATES

: FOR GOOD DATA  
: FOR BAD DATA

```

      . = 1000
001000 ICNT: 0 ; LH = ITERATION COUNT ; RH = TEST NO.
001002 ERRORS: 0 ; ERROR COUNT
001004 PCNT: 0.0 ; 2 WORD PASS COUNT
001010 LAD: 0 ; LOOP ADDRESS FOR SCOPE
001012 HLTADR: 0 ; ADDRESS OF LAST HLT INSTRUCTION EXECUTED
001014 FILCHR: 1000 ; FILCHR=0 (CHAR) ; FILCHR+1=2 (COUNT)
001016 TPS: 177564 ; OUTPUT STATUS REGISTER
001020 TKS: 177560
001022 TKB: 177562
001024 TPB: 177566 ; OUTPUT BUFFER
001026 SWR: 177570 ; SWITCH REGISTER
001030 DISPLAY: 177570 ; DISPLAY REGISTER

```

```

      . = 1100
001100 SAVBAD: 0 ; LOC FOR ILLEGAL FUNCTION CODE
001102 OBUFSV: 0 ; LOC OF OUTBUF

```

## :DISK I/O REGISTERS

```

001104 RSCS1: 172040 ; DISK CONTROL + STATUS REGISTER
001106 RSCS2: 172050 ; DISK CONTROL + STATUS REGISTER
001110 RSWC: 172042 ; WORD COUNT REGISTER
001112 RSBA: 172044 ; BUS ADDRESS
001114 RSDA: 172046 ; DISK ADDRESS (DESIRED ADDRESS)
001116 RSDS: 172052 ; DRIVE STATUS
001120 RSER: 172054 ; ERROR REG.
001122 RSAS: 172056 ; ATTENTION SUMMARY
001124 RSLA: 172060 ; LOOK AHEAD
001126 RSDB: 172062 ; DATA BUFFER REGISTER
001130 RSMR: 172064 ; MAINTENANCE REGISTER
001132 RSDT: 172066 ; DRIVE TYPE REGISTER
001134 RSVEC: 204 ; INTERRUPT VECTOR
001136 RSVCP: 206 ; INTERRUPT PRIORITY VECTOR
001140 RSCS1B: 172041 ; ODD BYTE ADD FOR CS1
001142 RSCS2B: 172051 ; ODD BYTE ADD FOR CS2
001144 RSWCB: 172043 ; ODD BYTE ADD FOR CW
001146 RSBAB: 172045 ; ODD BYTE ADD FOR BA

```

## :MEMORY MANAGEMENT REGISTER ASSIGNMENTS

```

SRO=177572
KIPARC=172340
KIPAR1=172342
KIPAR2=172344
KIPAR7=172356
KIPDR0=172300
KIPDR1=172302
KIPDR2=172304
KIPDR7=172316
RW=6
JP=00

```

```

:BIT ASSIGNMENTS FOR ERROR TYPEOUTS
:THE RS REGISTERS ARE DIVIDED INTO 3 GROUPS.
:CS1,CS2 AND ER ARE IN THE FIRST GROUP. THIS GROUP IS ALWAYS
:TYPED WITH EITHER OF THE OTHER GROUPS. AS,BA,DA,WC AND DS
:ARE IN THE SECOND GROUP. DT,DB,MR, AND LA ARE IN THE 3RD
:GROUP. YOU CAN NOT INTERMIX GROUP 2 OR 3. THEY HAVE
:TO BE TYPED SEPERATELY.
:EXAMPLE:  HLT !CS1,AS,BA
           HLT !CS1!DT!DB

```

```

CS1=1      :CONTROL AND STATUS 1
ER=2      :CONTROL AND STATUS 2
DA=4      :DESIRED ADD
WC=10     :WORD COUNT
BA=20     :BUS ADDRESS
DS=40     :DRIVE STATUS
AS=100    :ATTENTION SUMMARY
CS2=200   :CONTROL AND STATUS REG
LA=204    :LOOK AHEAD
DB=210    :DATA BUFFER
MR=220    :MAINTENANCE
DT=240    :DRIVE TYPE

```

```

:BIT ASSIGNMENTS FOR THE REGISTER BITS

```

```

TRE=40000 :TRANSFER ERROR CS1
C=100000  :SPECIAL CONDITIONS CS1
IR=100    :INPUT READY CS2
OR=200    :OUTPUT READY CS2
PGE=2000  :PROGRAM ERROR-CS2
NED=10000 :NON-EXISTENT DRIVE CS2
WCE=40000 :WRITE CHECK ERROR-CS2
DLT=100000:DATA LATE ERROR CS2
DRY=200   :DRIVE READY DS
PIP=20000 :POSITIONING IN PROGRESS DS
LBT=2000  :LAST BLOCK TRANSFER-DS
ERR=40000 :ERROR DS
ATA=100000:ATTENTION ACTIVE-DS
DAO=1000  :DISK OVERFLOW ERROR-ER
DCK=100000:DATA CHECK ERROR-ER
BAI=10    :BUS ADDR INCREMENT INHIBIT
IE=100    :INTERRUPT INABLE CS1

```

## :WORKING LOCATIONS

```

001150 RANNU: 146723 :RANDOM NUMBER PRIME
001152 UNNUM: 0 :UNIT CURRENTLY BEING TESTED
001154 UNITSV: 0 :SET BIT=UNIT ON BUS
001156 UNCMP: 0 :FOR COMPARING FOR # OF DEVICE
001160 ONCEE: 00 :DID WE TEST ANY DRIVES
001162 RS04DT: 0 :CLR IF RS03 SET IF RS04
001164 TIMSV: 0 :SAVE LOC FOR TIME
001166 AOB1: 0 :PORT SWITCH
WMP=4 :WRITE WRONG PARITY
MPRO=172100 :PARITY REG
001170 BPORTT: 0 :BUFFER ADDR FOR -B- PORT
001172 SAVEE: 0 :WORK LOC

```

## :DISCRIPTION OF ONCEE BITS

```

:BIT0 MEANS FOUND DRIVE
:BIT11 DO TKSEL TEST
:BIT12 TYPE COULD NOT FIND NED ONLY ONCE
:BIT13 TYPE NO MEM ON B PORT ONLY ONCE
:BIT14 0- DO WCE WITH 0 -1 DO WCE WITH 1
:BIT15 MEANS ERROR FOUND

```

## :RH11 WORK REGISTERS

:(CAN BE CHANGED IN ANY ROUTINE)

```

001174 WORK: 0
001176 WORK1: 0
001200 WORK2: 0

```

```

001202 BEGIN: MOV #500, SP :SET STACK TO *** 500 ***
001206 MOV #POWER, @#24 :SET UP PF VECTOR
001214 MOV #340, @#26 :LOCK OUT THE WORLD
001222 MOV #HL↑, @#30 :SET EMT VECTOR
001230 MOV #340, @#32 :LOCK UP
001236 MOV #TRAP, @#34 :SET TRAP VECTOR
001244 MOV #340, @#36 :LOCK UP
001252 CLR ICNT :INIT ICNT
001256 LAD :INIT LAD
001262 BIC #143777, ONCEE :CLEAR ONCEE
001270 MOV #OUTBUF, @GUFV :SAVE LOC OF OUTBUFFER
001276 BIT #BIT11, ONCEE :DO TKSEL TEST?
001304 BEQ +6 :NO
001306 JMP @TKSEL :YES
001312 SUSWR :SIZE FOR HDWR SWR

```

```

: NOW TEST FOR DRIVES
: *****
: TEST 1 TEST FOR DRIVES
: *****
001314  †ST1:  SCOPE

001316  MULTII: MOV      #8, R1          ; PUT 8 INTO R1 FOR COUNT
001322  CLR      @RSCS2        ; SET DEVICE TO ZERO
001326  TRY:   MOV      #7, @RSER      ; CAUSE AN ERROR +SETS BIT IN RSAS REG
001334  DEC      R1           ; DO A MAXIMUM OF 8 TIMES
001336  BEQ     DVNUM         ; TESTED FOR ALL DRIVES GET OUT
001340  INC     @RSCS2        ; INCREMENT DRIVE UNIT
001344  BR     TRY          ; REPEAT FOR NEXT DRIVE
001346  DVNUM: MOV     @RSAS, UNITSV ; SAVE
001354  MOV     #401, UNCOMP ; SETUP TO CMP WITH UNITSV
001362  MOV     #0, UNNUM     ; PUT 0 INTO UNIT NO.
001370  BIT     @BIT13, @SWR   ; INHIBIT TYPE OUT?
001376  BNE     STTEST      ; YES
001400  TYPE   ,.+2          ; .ASCIZ <15><12>"TESTING UNIT "
001424  BIC     @BIT15, ONCEE   ; CLEAR ERROR FLAG
001432  STTEST: BIT    UNCOMP, UNITSV ; IS THIS DRIVE ON THE SYSTEM
001440  BEQ     TRYNX        ; NO
001442  MOV     UNNUM, @RSCS2 ; YES PUT UNIT # INTO CS2
001450  CMP     #4, @RSDT    ; IS THIS A RS03LA?
001456  BNE     7$          ; NO
001460  MOV     #4, RS04DT   ; SETUP DRIVE TYPE FOR A LA DISK
001466  BR     1$           ; GET OUT
001470  7$:  CMP     #0, @RSDT ; IS THIS A RS03?
001476  BNE     2$          ; NO
001500  CLR     RS04DT      ; YES
001504  BR     1$           ; GET OUT
001506  2$:  CMP     #1, @RSDT ; IS THIS A RS03 4US?
001514  BNE     3$          ; NO
001516  CLR     RS04DT      ; RS03
001522  BR     1$           ; GET OUT
001524  3$:  CMP     #2, @RSDT ; IS THIS A RS04?
001532  BEQ     6$          ; YES
001534  CMP     #3, @RSDT   ; IS IT A RS04?
001542  BEQ     6$          ; YES
001544  BIC     UNCOMP, UNITSV ; CLEAR UNWANTED ATA BIT
001552  BR     TRYNX        ; GET A NEW NUMBER
001554  6$:  BIS     #-1, RS04DT ; YES RS04
001562  1$:  BIT     @BIT13, @SWR ; INHIBIT TYPE OUT?
001570  BNE     4$          ; YES
001572  BIT     @BIT15, ONCEE ; ANY ERRORS?
001600  BEQ     5$          ; NO
001602  TYPE   ,.+2          ; .ASCIZ <15><12><12>
001612  5$:  MOV     UNNUM, -(6) ; PUT UNNUM ON STACK
001616  TYPES  TYPE         ; TYPE STACK IN OCTAL - SUPRESS
001620  TYPE   ,40          ; TYPE SPACE
001624  4$:  BIC     @BIT15, ONCEE ; CLEAR ERROR FLAG
001632  BR     NOWGO         ; NOW TEST
    
```

```

001634 TRYNX: ASL UNCOMP :CHECK NEXT BIT FOR DRIVE
001640 BCS CHCKDV :DID WE TEST ANY REG?
001642 INC UNNUM :INC UNIT #
001646 BR STTEST :CHECK FOR NEXT DRIVE

001650 CHCKDV: BIT #BIT0,ONCEE :DID WE TEST ANY DRIVES?
001656 BNE DONEE :YES WE DID TEST A DRIVE
001660 MOV #100000,UNCOMP :NO DRIVES TESTED, COULD NOT SET
001666 CLR UNNUM :ANY AS BITS, THUS DEFAULTS TO
001672 BIT #BIT13,JSWR :INHIBIT TYPE OUT?
001700 BNE 4$ :YES
001702 MOV UNNUM,-(6) :PUT UNNUM ON STACK
001706 TYPES :TYPE STACK IN OCTAL - SUPRESS
001710 TYPE ,40 :TYPE SPACE
001714 TYPE ,.+2 :ASCIZ <15><12>"COULD NOT FIND DRIVE WILL TEST DRIVE 0 IF YOU CONTINUE"
002012 HALT :WAIT
002014 4$: BR NOWGO :TEST DRIVE 0
002016 DONEE: JMP DONE :GET OUT

```

:THIS TEST IS DESIGNED TO TEST THE ABILITY OF RESET  
:TO CLEAR ALL THE RH AND RS REGISTERS

```

002022 NOWGO: MOV #OUTBUF,OBUSV :SAVE LOC OF OUTBUFFER
002030 BIS #BIT0,ONCEE :SET FOUND DRIVE FLAG
002036 MOV TIMES,TIMSV :SAVE TIME
002044 MOV #1,TIMES :ONLY TEST ONCE

```

:\*\*\*\*\*  
:TEST 2 RESET TEST FOR REGISTERS  
:\*\*\*\*\*

```

002052 TST2: SCOPE
002054 MOV #340,JSPS :LOCK OUT INTERRUPTS
002062 MOV UNNUM,RS2 :GET UNIT #
002070 MOV #177776,RS1 :SET ALL
002076 MOV #177777,RSBA :POSSIBLE R/W
002104 MOV #177777,RSDA :BITS IN THESE REGISTERS
002112 MOV #177777,RSER
002120 MOV #177777,RSMR
002126 MOV #177777,RSWC
002134 MOV #177737,RS2
002142 RESET :CLEAR ALL BITS IN ALL REG.

```

:TEST RSCS2 FOR CLEARED BITS

```

002144 CMP #100,RS2 :DID THESE BITS GET CLEARED?
002152 BEQ +4 :YES
002154 HLT !CS2 : (417) SHOULD BE CLEARED IN CS2
002156 MOV UNNUM,RS2 :PUT # OF UNIT IN TEST IN CS2
002164 CMP #10600,RS5 :IS DPR AND MOL SET?
002172 BEQ +4 :YES
002174 HLT !DS :NO WHY NOT?

```

:TEST CONTROL AND STATUS REG 1

```

002176 CMP #4200,RS1 :DID THE READY BIT SET?
002204 BEQ +4 :YES
002206 HLT !CS1 :READY SHOULD BE SET

```

;TEST BUS ADDRESS REGISTER

002210	TST	QRSBA	; IS BA REG. CLEARED
002214	BEQ	.+4	; YES
002216	HLT	!BA	; SHOULD BE 0

;TEST DISK ADDRESS REGISTER

002220	TST	QRSDA	; IS DA CLEARED
002224	BEQ	.+4	; YES
002226	HLT	!DA	; SHOULD BE 0

;TEST ERROR REG RSER

002230	TST	QRSER	; DID RSER CLEAR?
002234	BEQ	.+4	; YES
002236	HLT	!ER	; BITS(157015) SHOULD BE CLEARED

;TEST RS MAINTENANCE REGISTER

002240	BIT	#77, QRSMR	; DID THESE BITS GET CLEARED
002246	BEQ	.+4	; YES
002250	HLT	!MR	; BITS(77) SHOULD BE 0

;TEST WC REG IT SHOULD NOT CHANGE

002252	CMP	#177777, QRSWC	; DID IT CHANGE?
002260	BEQ	.+4	; NO
002262	HLT	!WC	; RESET SHOULD NOT MODIFY RSWC

;TEST RSAS

002264	TST	QRSAS	; IS REG CLEAR
002270	BEQ	.+4	; YES
002272	HLT	!AS	; NO

.....  
:TEST 3 TEST CLEAR BIT IN CS2 ON ALL THE R/W BITS  
:TEST CLEAR BIT IN CS2 ON ALL THE R/W BITS  
.....

002274 TST3: SCOPE

002276 TTAGG: MOV 0340,0345 ;LOCK OUT INTERRUPTS  
002277 MOV UNNUM,03SCS2 ;GET UNIT #  
002278 MOV 043570,03XNUM ;GET ALL  
002279 MOV 017777,03XNUM ;POSSIBLE  
002280 MOV 017777,03XNUM ;REGISTERS  
002281 MOV 017777,03XNUM  
002282 MOV 017777,03XNUM  
002283 MOV 017777,03XNUM  
002284 MOV 020417,03SCS2  
002285 MOV 071,03S2  
002286 MOV 040,03S2

:CLEAR ALL BITS  
:DID THE RIGHT BITS CLEAR?  
:YES  
:(417) SHOULD BE CLEARED IN CS2  
:GET DRIVE NUMBER  
:DID ALL BITS GET CLEARED  
:YES  
:NO, ALL BITS SHOULD BE 0

:TEST BUS ADDRESS REGISTER

002432 TST 03RSBA ;IS BA REG. CLEARED  
002436 BEQ +4 ;YES  
002440 HLT !BA ;SHOULD BE 0

:TEST DISK ADDRESS REGISTER

002442 TST 03RSDA ;IS DA CLEARED  
002446 BEQ +4 ;YES  
002450 HLT !BA ;SHOULD BE 0

:TEST ERROR REG RSER

002452 TST 03RSER ;DID THESE BITS GET CLEARED  
002456 BEQ +4 ;YES  
002460 HLT !ER ;BITS(157015) SHOULD BE CLEARED

:TEST RS MAINTENANCE REGISTER

002462 BIT 077,03RSMR ;DID THESE BITS GET CLEARED  
002470 BEQ +4 ;YES  
002472 HLT !MR ;BITS(77) SHOULD BE 0

:TEST WC REG. IT SHOULD NOT CHANGE

002474 CMP 017777,03RWC ;DID WC CHANGE  
002502 BEQ +4 ;NO  
002504 HLT !WC ;WHY DID IT CHANGE?

\*\*\*\*\*  
:TEST 4 SET AND CLEAR ALL REGISTERS  
\*\*\*\*\*

002506 ↑TST4: SCOPE  
:CAN WE SET THE FUNCTION BITS IN THE RSCS1 REG.  
:BITS 7,6,5,4,3,2&1

002510 CLRDRK :CLEAR ALL RS REG  
002512 MOV TIMSV,TIMES :GET TIME  
002520 MOV #43576,RSCS1 :SET DISK FUNCTION BITS  
002526 CMP #7776,RSCS1 :ARE THESE BITS SET?  
002534 BEQ +4 :NO  
002536 HLT !CS1 :SHOULD = 7776  
002540 MOV #6724,RSCS1 :SET THESE BITS  
002546 CMP #6724,RSCS1 :DID THEY SET  
002554 BEQ +4 :YES  
002556 HLT !CS1 :SHOULD BE 6724  
002560 MOV #1052,RSCS1 :SET THESE BITS  
002566 CMP #5252,RSCS1 :ARE THEY =?  
002574 BEQ +4 :YES  
002576 HLT !CS1 :SHOULD = 5252

002600 TST5: SCOPE  
:CLEAR THE FUNCTION BITS

002602 MOV #43576,RSCS1 :SET DISK FUNCTION BITS  
002610 CLR RSCS1  
002614 CMP #4200,RSCS1 :IS THE READY BIT SET  
002622 BEQ +4 :YES  
002624 HLT !CS1 :RSCS1 SHOULD = 4200

\*\*\*\*\*  
:TEST 6 TEST RSCS2  
\*\*\*\*\*

002626 ↑TST6: SCOPE  
002630 RESET :CLEAR WORLD  
002632 CMP #100,RSCS2 :DID THEY CLEAR?  
002640 BEQ +4 :YES  
002642 HLT !CS2 :NO  
002644 MOV #21037,RSCS2 :SET BITS 21017  
002652 CMP #21137,RSCS2 :DID THESE BITS GET SET  
002660 BEQ 18 :YES  
002662 MOV RSCS2,BAD  
002666 MOV #21137,GOOD :WHAT CS2 SHOULD =  
002672 HLT :CS2 = BAD GOOD = CORRECT ANS

```

002674 18: MOV      #20025, @RSCS2      :SET THESE BITS
002702    CMP      #20125, @RSCS2      :DID THESE BITS GET SET
002710    BEQ      +4                      :YES
002712    HLT      !CS2                    :NO CS2 SHOULD = 20125
002714    MOV      #12, @RSCS2          :LOAD THESE BITS
002722    CMP      #112, @RSCS2         :DID THESE BITS GET SET IN CS2
002730    BEQ      +4                      :YES
002732    HLT      !CS2                    :BAD = CS2 GOOD = CORRECT ANS
002734    MOV      #-1, @RSCS2          :SET BITS
002742    CLR      @RSCS2                :CLEAR THEM
002746    CMP      #100, @RSCS2        :DID CLEAR WORK
002754    BEQ      +4                      :YES
002756    HLT      !CS2                    :R/W BITS DID NOT CLEAR
002760    MOV      UNNUM, @RSCS2        :GET UNIT #
002766    TST7:  SCOPE
    :CAN WE SET ALL THE RSBA BITS
    
```

```

002770    MOV      #177777, @RSBA        :SET THE BITS
002776    CMP      #177776, @RSBA        :DID THEY SET
003004    BEQ      +4                      :YES
003006    HLT      !BA                      :BITS 17776 SHOULD BE SET
003010    MOV      #125252, @RSBA        :SET THESE BITS
003016    CMP      #125252, @RSBA        :ARE THEY =
003024    BEQ      +4                      :YES
003026    HLT      !BA                      :SHOULD BE 125252
003030    MOV      #52524, @RSBA         :SET THESE BITS
003036    CMP      #52524, @RSBA        :ARE THEY =
003044    BEQ      +4                      :YES
003046    HLT      !BA                      :SHOULD BE 52524
    
```

003050 TST10: SCOPE  
 :FLOAT A 1 THROUGH RSBA

```

003052 FLOTBA: MOV      #2, GOOD          :GET A 2
003056    CLC                      :CLEAR CARRY
003060 18:  MOV      GOOD, @RSBA         :FLOAT NUMBER
003064    MOV      @RSBA, BAD          :GET BA
003070    CMP      GOOD, BAD           :COMPARE BA
003072    BEQ      +4                      :BA CORRECT
003074    HLT      !BAD=BA GOOD=CORRECT ANS
003076    ROL      GOOD              :ROTATE NUMBER
003100    BCC      18                    :LOOP TILL DONE
    
```

## E03

MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 63 SEQ 0029  
 DZRSBF.P11 27-SEP-77 15:12 TST6 TEST RSCS2

003102 TST11: SCOPE

:CLEAR THE RSBA REGISTER

003104 MOV #177777, @RSBA ;SET RSBA EQUAL TO ALL ONES  
 003112 CLR @RSBA  
 003116 TST @RSBA ;TEST FOR BIT0 SET IN RSBA (READ ONLY BIT)  
 003122 BEQ .+4 ;YES  
 003124 HLT !BA ;NO  
 003126 TST12: SCOPE

:CAN WE SET ALL BITS IN RSWC REGISTER

003130 MOV #177777, @RSWC ;SET WC BITS  
 003136 CMP #177777, @RSWC ;ARE ALL BITS SET  
 003144 BEQ .+4 ;YES  
 003146 HLT !WC ;NO  
 003150 MOV #125252, @RSWC ;SET THESE BITS  
 003156 CMP #125252, @RSWC ;ARE THEY =  
 003164 BEQ .+4 ;YES  
 003166 HLT !WC ;SHOULD BE 125252  
 003170 MOV #52525, @RSWC ;SET THESE BITS  
 003176 CMP #52525, @RSWC ;ARE THEY =  
 003204 BEQ .+4 ;YES  
 003206 HLT !WC ;SHOULD BE 152525  
 003210 TST13: SCOPE

:FLOAT A 1 THROUGH RSWC

003212 FLOTWC: MOV #1, GOOD ;GET A 1  
 003216 CLC ;CLEAR CARRY  
 003220 IS: MOV GOOD, @RSWC ;FLOAT NUMBER  
 003224 MOV @RSWC, BAD ;GET WC  
 003230 CMP GOOD, BAD ;COMPARE WC  
 003232 BEQ .+4 ;WC CORRECT  
 003234 HLT ;BAD=WC GOOD=CORRECT ANS  
 003236 ROL GOOD ;ROTATE NUMBER  
 003240 BCC IS ;LOOP TILL DONE

F03

003242 :CLEAR THE WORD COUNT REGISTER  
 TST14: SCOPE

003244 MOV #177777, @RSWC ;SET RSWC REGISTER EQUAL TO ALL ONES  
 003252 CLR @RSWC  
 003256 TST @RSWC ;DID ALL BITS GET CLEARED  
 003262 BEQ .+4 ;YES  
 003264 HLT !WC ;NO  
 003266 TST15: SCOPE

:CAN WE SET ALL THE BITS IN THE RSDA REGISTER.

003270 MOV #177777, @RSDA ;SET ALL BITS  
 003276 CMP #177777, @RSDA ;ARE THE BITS SET  
 003304 BEQ .+4 ;YES  
 003306 HLT !DA ;NO  
 003310 MOV #125252, @RSDA ;SET THESE BITS  
 003316 CMP #125252, @RSDA ;ARE THEY =  
 003324 BEQ .+4 ;YES  
 003326 HLT !DA ;SHOULD BE 125252  
 003330 MOV #52525, @RSDA ;SET THESE BITS  
 003336 CMP #52525, @RSDA ;ARE THEY =  
 003344 BEQ .+4 ;YES  
 003346 HLT !DA ;SHOULD BE 52525  
 003350 TST16: SCOPE

:FLOAT A 1 THROUGH RSDA

003352 FLOTDA: MOV #1, GOOD ;GET A 1  
 003356 CLC ;CLEAR CARRY  
 003360 IS: MOV GOOD, @RSDA ;FLOAT NUMBER  
 003364 MOV @RSDA, BAD ;GET DA  
 003370 CMP GOOD, BAD ;COMPARE DA  
 003372 BEQ .+4 ;DA CORRECT  
 003374 HLT ;BAD=DA GOOD=CORRECT ANS  
 003376 ROL GOOD ;ROTATE NUMBER  
 003400 BCC IS ;LOOP TILL DONE

003402 :CAN WE CLEAR THE RSDA REG.  
 TST17: SCOPE

003404 MOV #177777, @RSDA :SET RSDA TO ALL ONES  
 003412 CLR @RSDA  
 003416 TST @RSDA :TEST FOR ZERO RSDA  
 003422 BEQ +4 :YES  
 003424 HLT :DA :ANS SHOULD BE 0  
 003426 TST20: SCOPE

:SET AND CLEAR THE RSER REG.

00343C MOV #177777, @RSER :SET THESE BITS  
 003436 CMP #177017, @RSER :DID THEY SET  
 003444 BEQ +4 :YES  
 003446 HLT :ER :RSER SHOULD = 157017  
 003450 MOVB #1, @RSER :A MOVB INST  
 003456 CMP #1, @RSER :SHOULD MODIFY COMPLETE WC  
 003464 BEQ +4 :OK  
 003466 HLT :ER

003470 TST21: SCOPE

003472 MOV #52005, @RSER :SET THESE BITS  
 003500 CMP #52005, @RSER :DID THEY SET  
 003506 BEQ +4 :YES  
 003510 HLT :ER :ER SHOULD = 52005  
 003512 TST22: SCOPE

003514 MOV #125012, @RSER :SET THESE BITS  
 003522 CMP #125012, @RSER :DID THEY SET  
 003530 BEQ +4 :YES  
 003532 HLT :ER :ER SHOULD = 105012

# H03

```

003534 TST23: SCOPE
003536      MOV      #177017,DRSER      :SET THESE BITS
003544      CLR      DRSER             :CLEAR THEM
003550      TST      DRSER             :DID THEY CLEAR
003554      BEQ      +4               :YES
003556      HLT      !ER             :SHOULD = 0
003560 TST24: SCOPE
      :SET AND CLEAR RSMR
003562      MOV      #70,DRSMR        :SET THESE BITS
003570      MOV      DRSMR,WORK        :PUT INTO WORKABLE REG
003576      BIC      #177700,WORK    :CLEAR JUNK
003604      CMP      #70,WORK        :DID THEY SET
003612      BEQ      +4               :YES
003614      HLT      !MR             :SHOULD = 70
003616 TST25: SCOPE
003620      MOV      #70,DRSMR        :SET BITS
003626      CLR      DRSMR            :CLEAR THEM
003632      BIT      #77,DRSMR       :DID THEY CLEAR
003640      BEQ      +4               :YES
003642      HLT      !MR             :BITS (77) SHOULD = 0
003644 TST26: SCOPE
003646      MOV      #50,DRSMR       :SET BITS
003654      MOV      DRSMR,WORK      :PUT IN WORKABLE REG
003662      BIC      #177700,WORK    :CLEAR JUNK
003670      CMP      #50,WORK        :DID THESE BITS SET
003676      BEQ      +4               :YES
003700      HLT      !MR             :BITS (50) SHOULD BE SET
003702 TST27: SCOPE
003704      MOV      #20,DRSMR        :SET BITS
003712      MOV      DRSMR,WORK      :PUT INTO WORKABLE REG
003720      BIC      #177700,WORK    :CLEAR JUNK
003726      CMP      #20,WORK        :DID THEY SET
003734      BEQ      +4               :YES
003736      HLT      !MR             :MR SHOULD AT LEAST HAVE A 21
  
```

\*\*\*\*\*  
:TEST 30 LOAD RANDOM NUMBERS INTO RSWC, RSDA AND RSBA  
\*\*\*\*\*  
TST30: SCOPE

```

003740
003742 RANTS: MOV #1000,WORK :MAKE TABLE 1000 WDS LONG
003750 MOV #OUTBUF,R1 :GET STARTING LOC OF TABLE
003754 JSR RS,RANDOM :GENERATE #
003760 MOV #OUTBUF,R4 :SETUP FOR COMPARE
003764 MOV #LOP1,LAD :SETUP LOOP ADDR
003772 LOP1: MOV #1000,R3 :LOAD TEST COUNTER
003776 4$: DEC R3 :DONE WITH COMPARE?
004000 BEQ 1$ :YES
004002 MOV RSWC,R5 :GET WC ADDRESS
004006 MOV (R4),(R5) :LOAD WC
004010 CMP (R5),(R4)+ :IS IT CORRECT?
004012 BEQ 4$ :YES
004014 MOV @RSWC,BAD :GET BAD WC
004020 MOV -(R4),GOOD :GET GOOD ANS
004022 HLT :TYPE THEM OUT
004024 TST (R4)+ :UPDATE RANDOM NUMBER
004026 BR 4$ :CONT
004030 1$: MOV #OUTBUF,R4 :GET STARTING LOC OF TABLE
004034 MOV #LOP2,LAD :SETUP LOOP ADDR
004042 LOP2: MOV #1000,R3 :SETUP TEST COUNTER
004046 3$: DEC R3 :DONE YET?
004050 BEQ 1$ :YES
004052 MOV RSDA,R5 :LOAD DA ADDRESS INTO R5
004056 MOV (R4),(R5) :LOAD DA
004060 CMP (R5),(R4)+ :IS IT CORRECT?
004062 BEQ 3$ :YES
004064 MOV @RSDA,BAD :GET BAD DATA
004070 MOV -(R4),GOOD :GET GOOD DATA
004072 HLT :TYPE IT OUT
004074 TST (R4)+ :UPDATE RANDOM NUMBER
004076 BR 3$ :CONTINUE
004100 1$: MOV #OUTBUF,R4 :GET STARTING LOC OF TABLE
004104 MOV #LOP3,LAD :SETUP LOOP ADDR
004112 LOP3: MOV #1000,R3 :SETUP TEST COUNTER
004116 3$: DEC R3 :DONE YET?
004120 BEQ 2$ :YES
004122 MOV RSBA,R5 :LOAD ADDRESS OF BA INTO R5
004126 MOV (R4),(R5) :LOAD BA
004130 BIC #BIT0,(R4) :CLEAR BIT 0
004134 CMP (R5),(R4) :IS IT CORRECT?
004136 BEQ 3$ :YES
004140 MOV @RSBA,BAD :GET BAD DATA
004144 MOV (R4),GOOD :GET GOOD DATA
004146 HLT :TYPE IT OUT
004150 BR 1$ :GET OUT
004152 1$: TST (R4)+ :GET NEW NUMBER
004154 BR 3$ :CONTINUE
004156 2$: MCP

```

# J03

```

:*****
:TEST 31 TEST ODD BYTE INSTRUCTIONS ON CS1, CS2, WC AND BA
:*****
  
```

```

004160 TST31: SCOPE

004162 BITST: CLRDK ;CLEAR ALL RS REG
004164 MOV #3566, @RSCS1 ;LOAD CS1
004172 MOVB #5, @RSCS1B ;LOAD BIT
004200 CMP #6766, @RSCS1 ;DID IT LOAD?
004206 BEQ +4 ;YES
004210 HLT !CS1
004212 MOVB #32, @RSCS1
004220 CMP #6632, @RSCS1
004226 BEQ +4
004230 HLT !CS1 ;CS1 SHOULD = 6632

004232 TST32: SCOPE

004234 BITCS2: MOV UNNUM, @RSCS2 ;LOAD UNIT NUMBER
004242 BIS #177400, @RSCS2 ;LOAD ALL BITS
004250 CLR @RSCS2B ;CLR UPPER BYTE
004254 MOV UNNUM, GOOD ;GET UNIT NO.
004260 BIS #100, GOOD ;SET OR BIT
004264 MOV @RSCS2, BAD ;GET CS2
004270 CMP BAD, GOOD ;IS CS2 CORRECT?
004272 BEQ +4 ;YES
004274 HLT ;LOAD BYTE DID NOT WORK

004276 TST33: SCOPE

004300 BITWC: MOV #25252, @RSWC ;LOAD WC
004306 MOVB #377, @RSWCB ;LOAD BIT
004314 CMP #177652, @RSWC ;DID IT LOAD?
004322 BEQ +4 ;YES
004324 HLT !WC ;NO WC SHOULD =177652
004326 MOVB #123, @RSWC
004334 CMP #177523, @RSWC
004342 BEQ +4
004344 HLT !WC ;WC SHOULD = 177523

004346 TST34: SCOPE

004350 BITBA: MOV #25252, @RSBA ;LOAD DA
004356 MOVB #377, @RSBAB ;LOAD BIT
004364 CMP #177652, @RSBA ;DID IT LOAD?
004372 BEQ +4 ;YES
004374 HLT !BA ;DA SHOULD =177652
004376 MOVB #125, @RSBA
004404 CMP #177524, @RSBA
004412 BEQ +4
004414 HLT !BA ;BA SHOULD = 177525
004416 CLRDK ;CLEAR ALL RS REG
  
```

\*\*\*\*\*  
:TEST 35 TEST DATA LATE IN CS2  
\*\*\*\*\*

004420

↑TST35: SCOPE

;DO A READ FROM SILO: SHOULD GET DLT + TRE ERROR BECAUSE SILO IS EMPTY

004422  
004424  
004430  
004434  
004440  
004444  
004446  
004450  
004452  
004460  
004462  
004464  
004472  
004500  
004502  
004504  
004510  
004514  
004516  
004520

SILOB: CLRDK  
MOV @RSD8,BAD  
MOV @RSCS2,BAD  
MOV #100100,GOOD  
BIS UNNUM,GOOD  
CMP BAD,GOOD  
BEQ +4  
HLT :CS2  
CMP #144200,@RSCS1  
BEQ +4  
HLT :CS1  
MOV #TRE,@RSCS1  
BIT #140000,@RSCS1  
BEQ +4  
HLT :CS1  
MOV @RSCS2,BAD  
BIC #BIT15,GOOD  
CMP GOOD,BAD  
BEQ +4  
HLT :CS2

:CLEAR ALL RS REG  
:READ FROM EMPTY SILO  
:GET CS2  
:GET CORRECT ANS  
:FOR CS2  
:IS CS2 CORRECT?  
:YES  
:SHOULD HAVE DLT ERROR  
:DID SC AND TRE SET?  
:YES  
:SC AND TRE SHOULD BE SET  
:CLEAR ERROR BIT  
:DID SC + TRE CLEAR  
:YES  
:TRE AND SC SHOULD BE 0  
:GET CS2  
:GET CORRECT ANS  
:IS CS2 CORRECT?  
:YES  
:DLT SHOULD BE 0

\*\*\*\*\*  
:TEST 36 LOAD RSD8 WITH ALL ONES AND ALL ZEROS  
\*\*\*\*\*

004522

↑TST36: SCOPE

004524  
004526  
004532  
004540  
004546  
004552  
004556  
004562  
004564  
004566  
004572  
004574  
004576  
004600  
004604  
004606  
004610  
004612  
004616  
004622  
004624  
004626

ZEROME: CLRDK  
CLR @RSD8  
MOV #177777,@RSD8  
MOV #2000,WORK  
MOV #300,GOOD  
BIS UNNUM,GOOD  
25: MOV @RSCS2,BAD  
CMP GOOD,BAD  
BEQ 35  
DEC WORK  
BNE 25  
HLT :CS2  
35: CLR GOOD  
MOV @RSD8,BAD  
CMP GOOD,BAD  
BEQ +4  
HLT  
MOV #-1,GOOD  
MOV @RSD8,BAD  
CMP GOOD,BAD  
BEQ +4  
HLT

:CLEAR ALL RS REG  
:LOAD DB WITH ALL 0  
:LOAD DB WITH ALL ONES  
:TIME OUT ROUTINE  
:GET CORRECT FOR CS2  
:GET CS2  
:IS IT CORRECT?  
:YES  
:TO WAIT FOR OR  
:TO SET  
:OR SHOULD BE SET  
:LOAD BAD WITH DB  
:IS BAD CORRECT  
:YES  
:COULD NOT FLOAT 0 THROUGH DB  
:LOAD GOOD WITH ANS  
:GET DATA FROM DB  
:IS DB CORRECT  
:YES  
:BAD SHOULD = 1777777

## L03

MAINDEC-11-DZRSB-F RH11-RS03LA-RS03-RS04 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 70 SEQ 0036  
 DZRSBF.P11 27-SEP-77 15:12 TST36 LOAD RSDB WITH ALL ONES AND ALL ZEROS

004630 TST37: SCOPE  
 ;TEST FOR 66 LOCATIONS IN SILO PUT COUNT IN EVERY LOCATION

```

004632 SILO: CLRDK R1 ;CLEAR ALL RS REG
004634 CLR R1 ;CLEAR COUNTER
004636 1$: INC R1 ;INCREMENT COUNTER
004640 MOV R1, @RSDB ;LOAD SILO
004644 CMP #66., R1 ;LAST LOC. YET?
004650 BNE 1$ ;NO LOOP AGAIN
004652 MOV #200, GOOD ;GET CORRECT ANS FOR CS2
004656 BIS UNNUM, GOOD
004662 MOV @RSCS2, BAD ;GET CS2
004666 CMP GOOD, BAD ;IS CS2 CORRECT?
004670 BEQ .+4 ;YES
004672 HLT ;CS2 ;OR SHOULD BE 1
004674 CLR GOOD ;CLEAR LOCATION COUNTER
004676 2$: INC GOOD ;ADD 1 TO IT
004700 CMP #67., GOOD ;LAST LOC YET?
004704 BEQ 3$ ;YES
004706 MOV @RSDB, BAD ;GET LOC FROM DB
004712 CMP GOOD, BAD ;DO LOCATIONS MATCH?
004714 BEQ 2$ ;YES
004716 HLT ;CAN NOT MATCH 66 LOCATIONS
004720 3$: BIT #OR, @RSCS2 ;IS OR 0
004726 BEQ .+4 ;YES
004730 HLT ;OR SHOULD BE 0

```

;NOW PUT 67 WORDS INTO SILO AND CHECK FOR DLT ERROR

```

004732 CLR R1 ;CLEAR COUNTER
004734 4$: INC R1 ;ADD 1 TO COUNT
004736 MOV R1, @RSDB ;PUT INTO COUNTER
004742 CMP #67., R1 ;DONE YET?
004746 BEQ .+4 ;YES
004750 BR 4$ ;NO DO AGAIN
004752 BIT #DLT, @RSCS2 ;DID DATA LATE SET?
004760 BNE .+4 ;YES
004762 HLT ;DLT DID NOT SET

```

;DOES SILO CHANGE WITH 67TH WORD: IT SHOULD NOT

```

004764 MOV @RSDB, BAD ;GET 1ST WD FORM SILO
004770 MOV #1, GOOD ;CORRECT ANS OF SILO
004774 CMP GOOD, BAD ;IS SILO GOOD
004776 BEQ .+4 ;YES
005000 HLT ;SILO SHOULD NOT HAVE MOVED
005002 TST40: SCOPE

```

M03

:FLOAT A 1 AND A 0 THROUGH THE SILO

005004	SILOFL:	CLRDK		:CLEAR ALL RS REG
005006		CLC		:CLEAR CARRY TO FLOAT A 0
005010		MOV	#1,GOOD	:GET UP DATA FOR INPUT TO SILO
005014	1\$:	MOV	GOOD,RSDB	:LOAD DB
005020		ROL	GOOD	:SHIFT BIT
005022		BCS	.+4	:DONE YET SHIFTING?
005024		BR	1\$	:NO
005026		MOV	#-2,GOOD	:SET ALL ONES
005032		SEC		:SET CARRY TO ROL
005034	3\$:	MOV	GOOD,RSDB	:LOAD SILO
005040		ROL	GOOD	:SHIFT 0
005042		BCS	3\$	:LOOP TILL DONE

:NOW TEST OUTPUT

005044		CLC		:CLEAR CARRY
005046		MOV	#1,GOOD	:CORRECT ANS
005052	2\$:	MOV	RSDB,BAD	:GET DATA FROM DB
005056		CMP	GOOD,BAD	:IS DB DATA GOOD?
005060		BEQ	.+4	:YES
005062		HLT		:DB COULD NOT BUBBLE CORRECTLY
005064		ROL	GOOD	:SETUP FOR NEXT ANS
005066		BCS	.+4	:DONE YET?
005070		BR	2\$	:NO
005072		MOV	#-2,GOOD	:SETUP FOR ANS
005076	4\$:	MOV	RSDB,BAD	:GET DATA FROM DB
005102		CMP	GOOD,BAD	:IS IT CORRECT?
005104		BEQ	.+4	:YES
005106		HLT		:DB WRONG
005110		SEC		:SET CARRY TO ROL
005112		ROL	GOOD	:SETUP FOR NEXT ANS
005114		BCS	4\$	:LOOP TILL DONE

```

*****
:TEST 41 TEST NO-OP FUNCTION
*****

```

005116

TST41: SCOPE

005120

NOOP: CLRDK

:CLEAR ALL RS REG

005122

MOV #177777,OUTBUF

:DATA TO BE XFERED

005130

MOV @#0BUFSV,@RSBA

:SET UP CURRENT ADDRESS

005136

MOV #-1,@RSWC

:LOAD WC WITH -1

005144

MOV #1,@RSCSI

:DO NO-OP FUNCTION

005152

BIT #1,@RSCSI

:DID GO BIT CLEAR

005160

BEQ .+4

:YES

005162

HLT :CSI

:GO BIT SHOULD BE CLEARED

005164

TST @RSER

:DID ANY ERRORS OCCUR?

005170

BEQ .+4

:NO

005172

HLT :ER

:ALL ERROR BITS SHOULD BE 0

005174

CMP #-1,@RSWC

:DID WC MOVE?

005202

BEQ .+4

:NO

005204

HLT :WC

:WC SHOULD = 1777777

005206

TST @RSDA

:DID DA MOV

005212

BEQ .+4

:NO

005214

HLT :DA

:DA SHOULD =0

005216

CMP @#0BUFSV,@RSBA

:DID BA MOVE

005224

BEQ .+4

:NO

005226

HLT :BA

:BA MOVED

005230

BIT UNCMP,@RSAS

:AS SHOULD NOT SET ON

005236

BEQ .+4

:A NO-OP FUNCTION

005240

HLT :AS

:AS SET WHY?

\*\*\*\*\*  
:TEST 42 TEST NO-OP FUNCTION WITH ERROR BITS SET  
\*\*\*\*\*

005242 TST42: SCOPE

005244	NNOOP:	CLDRX	:CLEAR ALL RS REG
005246	MOV	#7,RSER	:LOAD ER
005248	BIT	UNCMP,RSAS	:IS ATA BIT SET?
005250	BNE	:+4	:YES
005252	HLT	:AS	:AS BIT SHOULD BE SET
005254	MOV	#177777,OUTBUF	:DATA TO BE XFERED
005256	MOV	#00BUFSV,RSBA	:SET UP CURRENT ADDRESS
005258	MOV	#-1,RSWC	:LOAD MC WITH -1
005260	MOV	#1,RSCSI	:DO NO-OP FUNCTION
005262	BIT	#1,RSCSI	:DID GO BIT CLEAR
005264	BRO	:+4	:YES
005266	HLT	:CSI	:GO BIT SHOULD BE CLEARED
005268	CTP	#150600,RSDS	:DID ERR BITS SET?
005270	BEG	:+4	:NO
005272	HLT	:DS	:ERR BIT SHOULD BE 0
005274	CTP	#-1,RSWC	:DID MC MOVE?
005276	BEG	:+4	:NO
005278	HLT	:MC	:MC SHOULD = 1777777
005280	TST	RSDA	:DID DA MOV
005282	BRO	:+4	:NO
005284	HLT	:DA	:DA SHOULD =0
005286	CTP	#00BUFSV,RSBA	:DID BA MOVE
005288	BEG	:+4	:NO
005290	HLT	:BA	:BA MOVED
005292	BIT	UNCMP,RSAS	:AS SHOULD BE SET
005294	BNE	:+4	:IS IT?
005296	HLT	:AS	:NO
005298	CTP	#7,RSER	:DID ER CHANGE?
005300	BEG	:+4	:NO
005302	HLT	:ER	:ER SHOULD NOT CHANGE

\*\*\*\*\*  
:TEST 43 TEST NO-OP FUNCTION CODE 21  
\*\*\*\*\*

005422 TST43: SCOPE

005422	NOOP21: CLROK		:CLEAR ALL RS REG
005423	MOV	#177777,OUTBUF	:DATA TO BE XFERED
005424	MOV	#00BUFSV,RSBA	:SET UP CURRENT ADDRESS
005425	MOV	#-1,RSWC	:LOAD MC WITH -1
005426	MOV	#21,RSCSI	:DO NO-OP FUNCTION
005427	BIT	#1,RSCSI	:DID GO BIT CLEAR
005428	BREQ	+4	:YES
005429	BT	CS1	:GO BIT SHOULD BE CLEARED
005430	TST	RSER	:DID ANY ERRORS OCCUR?
005431	BREQ	+4	:NO
005432	BT	ER	:ALL ERROR BITS SHOULD BE 0
005433	CTB	#-1,RSWC	:DID MC MOVE?
005434	BREQ	+4	:NO
005435	BT	MC	:MC SHOULD = 177777
005436	TST	RSDA	:DID DA MOV
005437	BREQ	+4	:NO
005438	BT	DA	:DA SHOULD =0
005439	CTB	#00BUFSV,RSBA	:DID BA MOVE
005440	BREQ	+4	:NO
005441	BT	BA	:BA MOVED
005442	BT	UNCMP,RSAS	:RS SHOULD NOT SET ON
005443	BREQ	+4	:A NO-OP FUNCTION
005444	BT	AS	:AS SET WHY?
005445	CTB	#4220,RSCSI	:IS CSI CORRECT?
005446	BREQ	+4	:YES
005447	BT	CS1	:CSI SHOULD = 4220

\*\*\*\*\*  
:TEST 44 TEST NO-OP FUNCTION CODE 21 WITH ERROR BITS SET  
\*\*\*\*\*

005560 TST44: SCOPE

005562	NNOP21: CLROK		:CLEAR ALL RS REG
005564	MOV	#7, @RSER	:LOAD ER
005572	BIT	UNCOMP, @RSAS	:IS ATA BIT SET?
005600	BNE	+4	:YES
005602	HLT	:AS	:AS BIT SHOULD BE SET
005604	MOV	#177777, @OUTBUF	:DATA TO BE XFERED
005612	MOV	@@0BUFSV, @RSBA	:SET UP CURRENT ADDRESS
005620	MOV	#-1, @RSWC	:LOAD WC WITH -1
005626	MOV	#21, @RSCS1	:DO NO-OP FUNCTION
005634	BIT	#1, @RSCS1	:DID GO BIT CLEAR
005642	BEQ	+4	:YES
005644	HLT	:CS1	:GO BIT SHOULD BE CLEARED
005646	CMP	#150600, @RSOS	:DID ERR BITS SET?
005654	BEQ	+4	:NO
005656	HLT	:DS	:ERR BIT SHOULD BE 0
005660	CMP	#-1, @RSWC	:DID WC MOVE?
005666	BEQ	+4	:NO
005670	HLT	:WC	:WC SHOULD = 1777777
005672	TST	@@RSOA	:DID DA MOV
005676	BEQ	+4	:NO
005700	HLT	:DA	:DA SHOULD =0
005702	CMP	@@0BUFSV, @RSBA	:DID BA MOVE
005710	BEQ	+4	:NO
005712	HLT	:BA	:BA MOVED
005714	BIT	UNCOMP, @RSAS	:AS SHOULD BE SET
005722	BNE	+4	:IS IT?
005724	HLT	:AS	:NO
005726	CMP	#7, @RSER	:DID ER CHANGE?
005734	BEQ	+4	:NO
005736	HLT	:ER	:ER SHOULD NOT CHANGE
005740	CMP	#104220, @RSCS1	:IS CS1 CORRECT?
005746	BEQ	+4	:YES
005750	HLT	:CS	:CS1 SHOULD = 104220

:\*\*\*\*\*  
:TEST 45 TEST DRIVE CLEAR FUNCTION WITH ERRORS SET  
:\*\*\*\*\*  
TST45: SCOPE  
:FIRST SET ALL R/W BITS IN DISK REG  
:DO DRIVE CLEAR-ALL R/W BITS SHOULD BE CLEARED

005752

005754	DRCLR: CLROK		: CLEAR ALL RS REG
005756	MOV	#177777, @RSWC	: LOAD RSWC
005764	MOV	#177777, @RSDA	: SET ALL POSSIBLE
005772	MOV	#177017, @RSER	: BITS IN DISK REG
006000	MOV	#70, @RSMR	: SET THESE BITS
006006	MOV	#11, @RSCS1	: SET DRIVE CLEAR
006014	MOV	@RSCS2, @BAD	: GET CS2 DATA
006020	BIC	#177640, @BAD	: CLEAR JUNK
006024	MOV	UNNUM, @GOOD	: GET DRIVE UNIT
006030	BIS	#100, @GOOD	: SET IR BIT
006034	CMP	GOOD, @BAD	: IS UNIT # THE SAME
006036	BEQ	.+4	: YES
006040	HLT		: UNIT # IN CS2 GOT MODIFIED
006042	TST	@RSDA	: DID DA CLEAR
006046	BEQ	.+4	: YES
006050	HLT	!DA	: DA SHOULD BE 0
006052	TST	@RSER	: DID ER CLEAR
006056	BEQ	.+4	: YES
006060	HLT	!ER	: ER SHOULD BE CLEARED
006062	MOV	@RSMR, @WORK	: GET MR REG
006070	BIC	#177707, @WORK	: CLEAR JUNK
006076	CMP	#70, @WORK	: IS 70 STILL SET IN MR?
006104	BEQ	.+4	: YES
006106	HLT	!MR	: BITS 70 SHOULD NOT CLEAR
006110	CMP	#4210, @RSCS1	: DID THESE BITS CLEAR?
006116	BEQ	.+4	: YES
006120	HLT	!CS1	: CS1 SHOULD =4210
006122	BIT	UNCOMP, @RSAS	: AS SHOULD NOT SET
006130	BEQ	.+4	: ON A DRIVE CLEAR FUN
006132	HLT	!AS	: WHY DID AS SET?
006134	CMP	#177777, @RSWC	: DID RSWC CHANGE?
006142	BEQ	.+4	: NO
006144	HLT	!WC	

F04

```

: DO ONE WORD WRITE
:*****
: TEST 46 EXECUTE THE ONE WORD WRITE
:*****
006146 TST46: SCOPE

006150 WRTST: CLDK          : CLEAR ALL RS REG
006152 MOV      #177777,OUTBUF : DATA TO BE X-FERED
006160 MOV      @#0BUFSV,@RSBA : SET UP CURRENT ADDRESS
006166 MOV      #-1,@RSWC       : SET WORD COUNT TO -1
006174 15: MOV      #60,@RSCS1 : SET FUNCTION WITH NO GO BIT
006202 CMP      #-1,@RSWC       : DID WC MOVE?
006210 BEQ      .+4           : NO
006212 HLT      !WC          : WC MOVED
006214 CMP      @#0BUFSV,@RSBA : DID RSBA MOVE?
006222 BEQ      .+4           : NO
006224 HLT      !BA          : BA MOVED
006226 BIS      @BIT0,@RSCS1    : SET GO BIT
006234 25: TSTB      @RSCS1    : TEST FOR RDY=0
006240 BPL      .+4           : RDY=0
006242 HLT      !CS1         : RDY SHOULD = 0
006244 JSR      PC,WAITRY     : WAIT FOR READY
006250 HLT      !CS1         : SHOULD = 260 RDY NEVER CAME UP
006252 CMP      #1,@RSDA      : IS RSDA CORRECT
006260 BEQ      .+4           : RSDA OK
006262 HLT      !DA          : SHOULD = 1 SHOULD INCREMENT
006264 35: CMP      #4260,@RSCS1 : IS ERROR FLAG SET?
006272 BEQ      .+4           : NO! X-FER OK
006274 45: HLT      !CS1!ER!DS!DA : ERROR DURING X-FER
006276 TST      @RSWC        : FETCH WORD COUNT
006302 BEQ      .+4           : WORD COUNT DID OVERFLOW
006304 HLT      !WC          : SHOULD = 0 FAILED TO INCREMENT
006306 CMP      #10600,@RSDS    : IS RSDS OK?
006314 BEQ      .+4           : YES
006316 HLT      !DS!DA        : NO
006320 MOV      UNNUM,GOOD      : GET UNIT #
006324 BIS      #100,GOOD      : SET IR BIT
006330 MOV      @RSCS2,BAD     : GET CS2
006334 CMP      GOOD,BAD      : IS CS2 CORRECT?
006336 BEQ      .+4           : YES
006340 HLT      !BAD           : BAD = CS2 GOOD IS CORRECT ANS
006342 MOV      @RSBA,BAD      : GET BA DATA
006346 MOV      @#0BUFSV,GOOD   : WHAT RSBA SHOULD EQUAL
006352 ADD      #2,GOOD       : UPDATE OUTBUFFER
006356 CMP      GOOD,BAD      : IS RSBA CORRECT
006360 BEQ      .+4           : YES
006362 HLT      !BA          : BA FAILED TO INCREMENT
006364 TST      @RSER         : DID ANY ERRORS SET?
006370 BEQ      .+4           : NO
006372 HLT      !DS
    
```

:TEST READ FUNCTION

:\*\*\*\*\*  
:TEST 47 EXECUTE THE ONE WORD READ  
:\*\*\*\*\*

006374 TST47: SCOPE

006376	RDST:	CLRDK	:	CLEAR ALL RS REG
006400		CLR	:	CLR TO READ INTO
006404		MOV	:	SET UP CURRENT ADDRESS
006412		MOV	:	SET WORD COUNT TO -1
006420	1S:	MOV	:	GO READ
006426	2S:	TSTB	:	TEST FOR BUSY=1
006432		BPL	:	BUSY SET
006434		HLT	:	BUSY NOT SET
006436		JSR	:	WAIT FOR READY
006442		HLT	:	TIMEOUT R0Y DID NOT SET
006444		CMP	:	WAS RSDA INCREMENTED BY 1
006452		BEQ	:	RSDA OK
006454		HLT	:	RSDA SHOULD CONTAIN A 1
006456	3S:	CMP	:	IS ERROR FLAG SET?
006464		BEQ	:	NO! X-FER OK
006466		HLT	:	RSCS1 SHOULD = 270
006470	4S:	TST	:	TEST WC
006474		BEQ	:	WORD COUNT DID OVERFLOW
006476		HLT	:	SHOULD = 0
006500		MOV	:	GET CORRECT
006504		BIS	:	ANS OF CS2
006510		MOV	:	GET CS2
006514		CMP	:	IS CS2 CORRECT?
006516		BEQ	:	YES
006520		HLT	:	GOOD = CORRECT ANS FOR CS2
006522		MOV	:	FETCH CURRENT ADDRESS
006526		MOV	:	WHAT RSBA SHOULD EQUAL
006532		ADD	:	UPDATE IT
006536		CMP	:	IS RSBA CORRECT
006540		BEQ	:	YES EXECUTE CONTINUE
006542		HLT	:	RSBA FAILED TO INCREMENT
006544		MOV	:	GET DATA READ FROM DISK
006550		MOV	:	GET CORRECT ANS
006554		CMP	:	IS OUTBUF CORRECT
006556		BEQ	:	YES
006560		HLT	:	GOOD=CORRECT ANS BAD=DATA READ FROM DISK

:\*\*\*\*\*  
:TEST 50 TEST WRITE CHECK  
:\*\*\*\*\*

006562 †TST50: SCOPE  
:DO A ONE WORD WRITE CHECK

:\* \* \*EXECUTE THE ONE WORD WRITE CHECK\* \* \*

006564	WRCKT:	CLRDK		:CLEAR ALL RS REG
006566		MOV	#177777,OUTBUF	:DATA TO BE X-FERED
006574		MOV	#0BUF5V,RSBA	:SET UP CURRENT ADDRESS
006602		MOV	#-1,RSWC	:SET WORD COUNT TO -1
006610	1\$:	MOV	#51,RS1	:GO WRITE CHECK
006616	2\$:	TSTB	RS1	:TEST FOR READY
006622		BPL	.+4	:NOT READY
006624		HLT	:CS1	:BUSY FAILED TO SET
006626	RSWCNT:	JSR	PC,WAITRY	:WAIT FOR READY
006632		HLT	:CS1	:BUSY FAILED TO CLEAR
006634		MOV	UNNUM,GOOD	:GET UNIT #
006640		BIS	#100,GOOD	:SET BIT IR
006644		MOV	RS2,BAD	:GET CS2
006650		CMP	GOOD,BAD	:IS CS2 CORRECT?
006652		BEQ	.+4	:YES
006654		HLT		:GOOD = CORRECT ANS FOR CS2
006656	2\$:	CMP	#4250,RS1	:ANY ERRORS?
006664		BEQ	.+4	:X-FER OK
006666		HLT	:DA!ER!DS	:ERROR DUR X-FER
006670	3\$:	CMP	#BIT0,RSDA	:WAS DAR INCREMENTED BY 1
006676		BEQ	.+4	:RSDA OK
006700		HLT	:DA	:DAR SHOULD = 1
006702		TST	RSWC	:TEST FOR OVERFLOW
006706		BEQ	.+4	:WORD COUNT DID OVERFLOW
006710		HLT	:WC	:SHOULD = 0
006712		MOV	RSBA,BAD	:FETCH CURRENT ADDRESS
006716		MOV	#0BUF5V,GOOD	:WHAT RSBA SHOULD EQUAL
006722		ADD	#2,GOOD	:UPDATE IT
006726		CMP	BAD,GOOD	:IS RSBA CORRECT
006730		BEQ	.+4	:YES EXECUTE CONTINUE
006732		HLT		:RSBA FAILED TO INCREMENT

: DO ONE WORD WRITE ON -B- PORT  
 : IF A 1 MD TRANSFER KEEPS SETTING NEM PROGRAM WILL GO AND UPDATE  
 : ADDRESS (OBUFSV) ON -B- PORT BY 4K AND TRY TRANSFER AGAIN UNTILL IT  
 : REACHES 2BK. IF NO TRANSFER IT THEN SKIPS WRITE.  
 : READ AND WRITE CHECK ON -B- PORT  
 : TO INHIBIT OBUFSV FROM CHANGING SET BIT 12

\*\*\*\*\*  
 : TEST 51 EXECUTE THE ONE WORD WRITE ON -B- PORT  
 \*\*\*\*\*  
 †TST51: SCOPE

006734

006736	WRSTB:	CLROK	:	CLEAR ALL RS REG
006740		MOV	2#OBUFSV,WORK	: GET LOC OF RSBA TO LOAD
006746		MOV	2#177777,WORK	: DATA TO BE X-FERED
006754		MOV	2#OBUFSV,RSBA	: SET UP CURRENT ADDRESS
006762		MOV	2#-1,RSWC	: SET WORD COUNT TO -1
006770		MOV	2#2061,RSCS1	: TEST B PORT
00677E	2\$:	TSTB	2#RSCS1	: TEST FOR RDY=0
007002		BPL	.+4	: ROY=0
007004		HLT	!CS1	: ROY SHOULD = 0
007006		JSR	PC,WAITRY	: WAIT FOR READY
007012		HLT	!CS1	: SHOULD = 260 ROY NEVER CAME JP
007014		BIT	2#BIT12,2SWR	: INHIBIT ADDRESS?
007022		BNE	3\$	: YES
007024		BIT	2#BIT11,2RSCS2	: DID NEM SET?
007032		BEQ	3\$	: NO
007034		JMP	FINDM	: GO FIND MEMORY ON PORT B
007040	3\$:	CMP	2#1,2RSDA	: IS RSDA CORRECT
007046		BFG	.+4	: RSDA OK
007050		HLT	!DA	: SHOULD = 1 SHOULD INCREMENT
007052		CMP	2#6260,2RSCS1	: IS CS1 CORRECT?
007060		BEQ	4\$	: YES
007062		HLT	!CS1	
007064	4\$:	TST	2#RSWC	: FETCH WORD COUNT
007070		BEQ	.+4	: WORD COUNT DID OVERFLOW
007072		HLT	!WC	: SHOULD = 0 FAILED TO INCREMENT
007074		CMP	2#10600,2RSDS	: IS RSDS OK?
007102		BEQ	.+4	: YES
007104		HLT	!DS!DA	: NO
007106		MOV	UNNUM,GOOD	: GET UNIT #
007112		BIS	2#100,GOOD	: SET IR BIT
007116		MOV	2#RSCS2,BAD	: GET CS2
007122		CMP	GOOD,BAD	: IS CS2 CORRECT?
007124		BEQ	.+4	: YES
007126		HLT		: BAD = CS2 GOOD IS CORRECT ANS
007130		MOV	2#RSBA,BAD	: GET BA DATA
007134		MOV	2#OBUFSV,GOOD	: WHAT RSBA SHOULD EQUAL
007140		ADD	2#2,GOOD	: UPDATE OUTBUFFER
007144		CMP	GOOD,BAD	: IS RSBA CORRECT
007146		BEQ	.+4	: YES
007150		HLT		: BA FAILED TO INCREMENT

\*\*\*\*\*  
:TEST 52 EXECUTE THE ONE WORD READ ON -B- PORT  
\*\*\*\*\*

007152 †TST52: SCOPE

007154	RDTSTB: CLRDK		: CLEAR ALL RS REG
007156	CLR	WORK	: CLR TO READ INTO
007162	MOV	2#0BUFSV, 2RSBA	: SET UP CURRENT ADDRESS
007170	MOV	#-1, 2RSWC	: SET WORD COUNT TO -1
007176	MOV	2071, 2RSCS1	: B PORT
007204	2S: TSTB	2RSCS1	: TEST FOR BUSY=1
007210	BPL	.+4	: BUSY SET
007212	HLT	:CS1	: BUSY NOT SET
007214	JSR	PC, WAITRY	: WAIT FOR READY
007220	HLT	:CS1	: TIMEOUT RDY DID NOT SET
007222	CMP	#BITO, 2RSDA	: WAS RSDA INCREMENTED BY 1
007230	BEG	.+4	: RSDA OK
007232	HLT	:DA!ER!DS	: RSDA SHOULD CONTAIN A 1
007234	CMP	#6270, 2RSCS1	: TST B PORT
007242	BEG	4S	: OK
007244	HLT	:CS1	: CS1 SHOULD = 6270
007246	4S: TST	2RSWC	: TEST WC
007252	BEG	.+4	: WORD COUNT DID OVERFLOW
007254	HLT	:WC	: SHOULD = 0
007256	MOV	UNNUM, GOOD	: GET CORRECT
007262	BIS	#100, GOOD	: ANS OF CS2
007266	MOV	2RSCS2, BAD	: GET CS2
007272	CMP	GOOD, BAD	: IS CS2 CORRECT?
007274	BEG	.+4	: YES
007276	HLT		: GOOD = CORRECT ANS FOR CS2
007300	MOV	2RSBA, BAD	: FETCH CURRENT ADDRESS
007304	MOV	2#0BUFSV, GOOD	: WHAT RSBA SHOULD EQUAL
007310	ADD	#2, GOOD	: UPDATE IT
007314	CMP	BAD, GOOD	: IS RSBA CORRECT
007316	BEG	.+4	: YES EXECUTE CONTINUE
007320	HLT		: RSBA FAILED TO INCREMENT
007322	MOV	2#0BUFSV, WORK	: GET DATA READ FROM DISK
007330	MOV	2WORK, BAD	
007334	MOV	#-1, GOOD	: GET CORRECT ANS
007340	CMP	GOOD, BAD	: IS OUTBUF CORRECT
007342	BEG	.+4	: YES
007344	HLT		: GOOD=CORRECT ANS BAD=DATA REAC FROM DISK

\*\*\*\*\*  
:TEST 53 TEST ITE CHECK ON -B- PORT  
\*\*\*\*\*

007346 TST53: SCOPE

007350	WRCKTB: CLRDK		:CLEAR ALL RS REG
007352	MOV	@#0BUFSV,WORK	:GET LOC FOR
007360	MOV	@177777,@WORK	:DATA TO BE X-FERED
007366	MOV	@#0BUFSV,@RSBA	:SET UP CURRENT ADDRESS
007374	MOV	@-1,@RSWC	:SET WORD COUNT TO -1
007402	MOV	@2051,@RSCS1	:B PORT
007410	2\$: TSTB	@RSCS1	:TEST FOR READY
007414	BPL	.+4	:NOT READY
007416	HLT	:CS1	:BUSY FAILED TO SET
007420	JSR	PC,WAIRY	:WAIT FOR READY
007424	HLT	:CS1	:BUSY FAILED TO CLEAR
007426	MOV	UNNUM,GOOD	:GET UNIT #
007432	BIS	@100,GOOD	:SET BIT IR
007436	MOV	@RSCS2,BAD	:GET CS2
007442	CMP	GOOD,BAD	:IS CS2 CORRECT?
007444	BEQ	.+4	:YES
007446	HLT		:GOOD = CORRECT ANS FOR CS2
007450	CMP	@6250,@RSCS1	:IS CS1 CORRECT?
007456	BEQ	3\$	:YES
007460	HLT	:CS1	:CS1 SHOULD = 6250
007462	3\$: CMP	@BIT0,@RSDA	:WAS DAR INCREMENTED BY :
007470	BEQ	.+4	:RSDA OK
007472	HLT	:DA	:DAR SHOULD = 1
007474	TST	@RSWC	:TEST FOR OVERFLOW
007500	BEQ	.+4	:WORD COUNT DID OVERFLOW
007502	HLT	:WC	:SHOULD = 0
007504	MOV	@RSBA,BAD	:FETCH CURRENT ADDRESS
007510	MOV	@#0BUFSV,GOOD	:WHAT RSBA SHOULD EQUAL
007514	ADD	@2,GOOD	:UPDATE IT
007520	CMP	BAD,GOOD	:IS RSBA CORRECT
007522	BEQ	.+4	:YES EXECUTE CONTINUE
007524	HLT		:RSBA FAILED TO INCREMENT

```
007526 NXM: MOV OBUFSV,BPORTT ;SAVE -B- PORT BUFFER
007534 MOV #OUTBUF,OBUSV ;RESTORE OBUFSV
```

```
:DESELECT THEN SELECT UNIT NUMBER IN RSCS2 CHECK TIMING
:*****
:TEST 54 DESELECT THEN SELECT UNIT NUMBER TIMING TEST
:*****
```

```
007542 †TST54: SCOPE
```

```
007544 UNITST: CLRDK ;CLEAR ALL RS REG
007546 CLR R4 ;CLEAR R4
007550 CMP R4,UNNUM ;IS THIS CORRECT UNIT #?
007554 BNE 3$ ;NO THEN USE IT
007556 INC R4 ;GET WRONG DRIVE
007560 3$: MOV #177777,OUTBUF ;DATA TO BE X-FERED
007566 MOV #OBUSV,RSBA ;SET UP CURRENT ADDRESS
007574 MOV #-1,RSWC ;SET WORD COUNT TO -1
007602 MOV #61,R3 ;GET WRITE FUNCTION
007606 MOV UNNUM,RS ;GET CORRECT UNIT #
007612 MOV #172040,R1 ;GET CS1 REG
007616 MOV R4,10(R1) ;LOAD WRONG UNIT # INTO CS2
007622 NOP ;WAIT FOR DRIVE TO SETTLE
007624 1$: MOV RS,10(R1) ;LOAD CORRECT UNIT #
007630 MOV R3,(R1) ;LOAD FUNCTION IN CS1
007632 JSR PC,WAITRY ;WAIT FOR READY
007636 HLT !CS1 ;SHOULD = 260 RDY NEVER CAME UP
007640 CMP #4260,RSRCS1 ;IS ERROR FLAG SET?
007646 BEQ .+4 ;NO! X-FER OK
007650 HLT !CS1!ER!DS!DA ;ERROR DURING X-FER
007652 CMP #1,RSRSDA ;IS RSDA CORRECT
007660 BEQ .+4 ;RSDA OK
007662 HLT !DA ;SHOULD = 1 SHOULD INCREMENT
007664 TST RSWC ;FETCH WORD COUNT
007670 BEQ .+4 ;WORD COUNT DID OVERFLOW
007672 HLT !WC ;SHOULD = 0 FAILED TO INCREMENT
007674 CMP #10600,RSRSDS ;IS RSDS OK?
007702 BEQ .+4 ;YES
007704 HLT !DS!DA ;NO
007706 MOV UNNUM,GOOD ;GET UNIT #
007712 BIS #100,GOOD ;SET IR BIT
007716 MOV RSCS2,BAD ;GET CS2
007722 CMP GOOD,BAD ;IS CS2 CORRECT?
007724 BEQ .+4 ;YES
007726 HLT ;BAD = CS2 GOOD IS CORRECT ANS
007730 MOV RRSBA,BAD ;GET BA DATA
007734 MOV #OBUSV,GOOD ;WHAT RSBA SHOULD EQUAL
007740 ADD #2,GOOD ;UPDATE OUTBUFFER
007744 CMP GOOD,BAD ;IS RSBA CORRECT
007746 BEQ .+4 ;YES
007750 HLT ;BA FAILED TO INCREMENT
```

:TEST CURRENT ADDRESS INHIBIT-BAI IN RSCS2  
:DO A ONE WORD WRITE AND SEE  
:IF RSBA INCREMENTED AFTER THE X-FER

:\*\*\*\*\*  
:TEST 55 TEST BAI IN RSCS2  
:\*\*\*\*\*

007752

TST55: SCOPE

007754

BAITST: CLRDK

007756

MOV

@#0BUFSV,@RSBA

:CLEAR ALL RS REG

007764

MOV

#-1,@R5WC

:SET UP CURRENT ADDR

007772

BIS

#BAI,@RSCS2

:SET WORD COUNT TO -1

010000

MOV

#51,@RSCS1

:SET BAI BIT

010006

JSR

PC,WAITRY

:WRITE

010012

HLT

!CS1

:WAIT FOR READY

010014

IS:

MOV

@#0BUFSV,GOOD

:RDY DID NOT SET

010020

MOV

@RSBA,BAD

:WHAT RSBA SHOULD BE

010024

CMP

GOOD,BAD

:WHAT RSBA IS

010026

BEQ

.+4

:COMPARE

010030

HLT

:YES  
:BAD=OUTBUF GOOD = CORRECT ANS

010032

TST

@RSER

:ANY ERRORS?

010036

BEQ

.+4

:NO

010040

HLT

!DS

:YES

010042

CLRDK

:CLEAR ALL RS REG

010044

BIT

#BAI,@RSCS2

:DID BAI CLEAR?

010052

BEQ

.+4

:YES

010054

HLT

!ER

:BAI DID NOT SET

\*\*\*\*\*  
:TEST 56 TEST NON-EXISTENT MEMORY ERROR BIT IN CS2  
\*\*\*\*\*

010056 TST56: SCOPE

010060	NXMTSM: CLRDK		: CLEAR ALL RS REG
010062	BIS	#BAI, @RSCS2	: SET BAI BIT
010070	MOV	#-200, @RSWC	: SET UP WORD COUNT
010076	MOV	#173000, @RSBA	: SET UP CURRENT ADDRESS
010104	MOV	#1471, @RSCS1	: READ AND LOAD A16 +A17 FOR 18 BIT ADDRESS
010112	JSR	PC, WAITRY	: WAIT FOR READY
010116	HLT	!DS	: READY NEVER CAME UP
010120	TSTNEM: MOV	UNNUM, GOOD	: GET UNIT NO.
010124	BIS	#4310, GOOD	: SET BAI+OR BITS
010130	MOV	@RSCS2, BAD	: GET CS2
010134	CMP	GOOD, BAD	: IS CS2 CORRECT?
010136	BEQ	+.4	: YES
010140	HLT		: BAD=CS2 GOOD=CORRECT ANS FOR CS2
010142	CMP	#145670, @RSCS1	: DID TRE SET?
010150	BEQ	+.4	: YES
010152	HLT	!CS1	: TRE SHOULD SET BECAUSE OF NEM
010154	MOV	#TRE, @RSCS1	: CLEAR TRE
010162	MOV	@RSCS2, BAD	: GET CS2
010166	MOV	UNNUM, GOOD	: GET DRIVE
010172	BIS	#310, GOOD	: SET IR
010176	CMP	GOOD, BAD	: IS CS2 CORRECT?
010200	BEQ	+.4	: YES
010202	HLT	!CS2	: CS2=BAD GOOD IS CORRECT ANS FOR CS2

\*\*\*\*\*  
:TEST 57 TEST BLOCK SEARCH FUNCTION, PIP AND DRY BIT AND ADDR. CONF BIT  
\*\*\*\*\*

010204 TST57: SCOPE

010206	BLOCK: CLRDK		: CLEAR ALL RS REG
010210	MOV	#32, @RSDA	: DO A SEARCH FOR SECTOR 32
010216	MOV	@#0BUFSV, @RSBA	: LOAD REGS. TO MAKE
010224	MOV	#-1, @RSWC	: SURE THEY DO NOT CHANGE
010232	CLR	WORK	: SETUP FOR TIMEOUT ROUTINE
010236	4\$: BIT	#1000, @RSMR	: WAIT FOR DISK TO
010244	BNE	3\$	: REACH SECTOR 32
010246	DEC	WORK	: TIME OUT
010252	BNE	4\$	: ROUTINE
010254	HLT	!MR	: COULD NOT FIND SECTOR 32



\*\*\*\*\*  
:TEST 60 ILLEGAL FUNCTION CODE TEST CODE 3 TO 51  
\*\*\*\*\*

010504 TST60: SCOPE

:TEST ILF BIT IN RSER AND ERR BIT IN RSDS  
:ALSO CHECKS TO SEE IF WC, BA, OR DA GOT MODIFIED  
:IF WISHING TO LOOP ON ONE FUNCTION ONLY, LOAD  
:FUNCTION INTO LOCATION ILLTAB: AND 0 IN FOLLOWING LOCATION

010506	ILLS1:	MOV	TIMES, TMSV	:SAVE LOOP COUNT
010514		MOV	#10, TIMES	:LOOP TEN TIMES
010522		CLRDK		:CLEAR ALL RS REG
010524	18:	MOV	#ILLTAB, R3	:GET STARTING ADD OF TABLE
010530	38:	MOV	(R3)+, BAD	:GET ILL FUN
010532		BEG	ILFDN	:DONE GET OUT
010542		MOV	#080, RSV, RSBA	:SET UP REGS
010544		MOV	#-1, RSAC	:TO CHECK FOR CHANGE
010550	28:	MOV	BAD, RSCS1	:DO ILLEGAL FUNCTION
010554		BIC	#10, BAD	:CLEAR GO BIT
010560		MOV	#0, GOOD	:MOV ILLEGAL FUN INTO GOOD
010562	68:	TSTB	RSAC	:ROY SET?
010566		BPL	68	:NO
010570		BIS	#104200, GOOD	:SET ERROR BITS
010574	48:	MOV	RSAC, BAD	:PUT CS1 INTO BAD
010600		MOV	GOOD, BAD	:IS CS1 CORRECT?
010602		BEG	.+4	:YES
010604		HLT		:GOOD IS WHAT CS1 SHOULD =BAD=CS1
010606		MOV	#1, RSER	:DID ILF SET?
010614		BEG	.+4	:YES
010616		HLT	:CS1!ER!DS	:ILF DID NOT SET
010620		MOV	#150600, RSDS	:IS DS GOOD?
010622		BEG	.+4	:YES
010630		HLT	:CS1!ER!DS	:ERR DID NOT SET
010632		MOV	RSAC, BAD	:GET CS2
010636		MOV	UNNUM, GOOD	:GET UNIT #
010642		BIS	#100, GOOD	:SET IR BIT
010646		MOV	GOOD, BAD	:IS CS2 CORRECT?
010650		BEG	.+4	:YES
010652		HLT		:GOOD = CORRECT ANS FOR CS2
010654		MOV	UNCMP, GOOD	:GET CORRECT DRIVE
010660		BIC	#177400, GOOD	:CLEAR UNWANTED BITS
010664		MOV	RSAS, #0	:GET RSAS REG
010670		MOV	BAD, GOOD	:DID CORRECT UNIT ANSWER?
010672		BEG	.+4	:YES
010674		HLT	:AS	:NO WRONG DRIVE ANSWERED

```

010676      CMP      280BUFSV,2RSBA      :DID BA MOVE
010704      BEQ      +4                  :NO
010706      HLT      :CS1:BA           :BA MOVED ON AN ILLEGAL FUNCTION
010710      CMP      #-1,2RSWC        :DID WC MOVE?
010716      BEQ      +4                  :NO
010720      HLT      :CS1:WC         :WC MOVED
010722      TST      2RS0A           :DID DA MOVE
010726      BRD      +4                  :NO
010730      HLT      :CS1:DA         :DA MOVED
010732      CLDK      :CLEAR ALL ERRORS
010734      TST      2RSER           :DID ERRORS CLEAR
010740      BRD      +4                  :YES
010742      HLT      :IF DID NOT CLEAR
010744      CMP      #4200,2RSCS1    :DID ERRORS IN CS1 CLEAR
010752      BEQ      +4                  :YES
010754      HLT      :DS
010756      JMP      3$              :CONTINUE UNTIL DONE
010762      ILFDN:                    :DONE WITH ILLEGAL FUNCTION TEST

```

```

:*****
:TEST 61      ILLEGAL FUNCTION CODE TEST CODE 53 TO 77
:*****

```

```

010762      TST61: SCOPE
:TEST ILF BIT IN RSER AND ERR BIT IN RSDS
:ALSO CHECKS TO SEE IF WC, BA, OR DA GOT MODIFIED
:IF WISHING TO LOOP ON ONE FUNCTION ONLY, LOAD
:FUNCTION INTO LOCATION ILFTB2: AND 0 IN FOLLOWING LOCATION

```

```

010764      ILLFUN: CLDK      :CLEAR ALL RS REG
010766      1$:      MOV      #ILFTB2,R3 :GET TABLE OF ILL FUNS.
010772      3$:      CLR      WORK      :CLEAR WORK
010776      MOV      (R3)+,BAD         :GET ILL FUN
011000      BEQ      ILFDN           :DONE GET OUT
011002      MOV      280BUFSV,2RSBA   :SET UP REGS.
011010      MOV      #-1,2RSWC        :TO CHECK FOR CHANGE
011016      MOV      BAD,WORK1        :SHOULD WE TEST
011022      BIC      #17707,WORK1     :BA AND WC
011030      CMP      #60,WORK1        :TO INC
011036      BNE      2$              :NO
011040      MOV      #7,WORK          :YES
011046      2$:      MOV      BAD,2RSCS1 :DO ILLEGAL FUNCTION
011052      10$:     BIC      #BIT0,BAD :CLEAR GO BIT
011056      MOV      BAD,GOOD        :MOV ILLEGAL FUN INTO GOOD

```

E05

```

011060 6S: TSTB 2RSCS1 :ROY SET?
011064 BPL 6S :NO
011066 BIS #144200,GOOD :SET ERROR BITS
011072 MOV 2RSCS1,BAD :PUT CS1 INTO BAD
011076 CMP GOOD,BAD :IS CS1 CORRECT?
011100 BEQ .+4 :YES
011104 HLT :GOOD IS WHAT CS1 SHOULD =BAD=CS1
011108 CMP #1,2RSER :DID ILF SET?
011112 BEQ .+4 :YES
011116 HLT :CS1!ER!DS :ILF DID NOT SET
011120 CMP #150600,2RSDS :IS DS GOOD?
011124 BEQ .+4 :YES
011128 HLT :CS1!ER!DS :ERR DID NOT SET
011132 TST 2RSDA :DID DA MOVE?
011136 BEQ .+4 :NO
011140 HLT :CS1!DA :DA MOVED
011144 TST WORK :IS THIS AN ILL WRITE FUN?
011148 BNE 11S :YES
011152 MOV 2RSCS2,BAD :GET CS2
011156 MOV UNNUM,GOOD :GET UNIT #
011160 BIS #1100,GOOD :SET IR BIT
011164 CMP GOOD,BAD :IS CS2 CORRECT?
011168 BEQ .+4 :YES
011172 HLT :GOOD = CORRECT ANS FOR CS2
011176 CMP 2#0BUFSV,2RSBA :DID BA MOVE
011180 BEQ .+4 :NO
011184 HLT :CS1!BA :BA MOVED ON AN ILLEGAL FUNCTION
011188 CMP #1,2RSWC :DID WC MOVE?
011192 BEQ .+4 :NO
011196 HLT :CS1!WC :WC MOVED
011200 JMP 4S :CONTINUE UNTIL DONE
011204 11S: MOV 2RSCS2,BAD :GET CS2
011208 MOV UNNUM,GOOD :GET UNIT #
011212 BIS #1300,GOOD :SET IR BIT
011216 CMP GOOD,BAD :IS CS2 CORRECT?
011220 BEQ .+4 :YES
011224 HLT :GOOD = CORRECT ANS FOR CS2
011228 MOV 2#0BUFSV,WORK :GET BUFFER ADDR.
011232 ADD #2,WORK :UPDATE IT
011236 CMP WORK,2RSBA :DID BA MOVE
011240 BEQ .+4 :YES
011244 HLT :CS1!BA :BA MOVED ON AN ILLEGAL FUNCTION
011248 CMP #0,2RSWC :DID WC MOVE?
011252 BEQ .+4 :NO
011256 HLT :CS1!WC :WC MOVED
011260 4S: CLDRK :CLEAR ALL ERRORS
011264 CMP #4200,2RSCS1 :DID ERRORS CLEAR
011268 BEQ .+4 :YES
011272 HLT :DS :NO
011276 TST 2RSER :DID ERROR CLEAR
011280 BEQ .+4 :NO
011284 HLT :DS :YES
011288 JMP 3S :CONTINUE UNTIL DONE
011292 1.LFDNE: :DONE WITH ILLEGAL FUNCTION TEST
    
```

F05

MAINDEC-11-DZRSB-F  
DZRSBF.P11

27-SEP-77 15:12

RH11-R503LA-R503-R504

TST62 TEST ILLEGAL FUNCTION CODE 67

BASIC FUNCTION DIAGNOSTIC MACY11 30(1046)

28-SEP-77

10:04

PAGE 90

SEG 0056

:\*\*\*\*\*  
:TEST 62 TEST ILLEGAL FUNCTION CODE 67  
:\*\*\*\*\*

```

011332 TST62: SCOPE
011334 ILF67: CLDK          :CLEAR ALL RS REG
011336 MOV      # -1,RSWC    :SET WC TO -1
011344 MOV      @OBUF SV,WORK :GET OUTBUF ADD.
011352 ADD      #2,WORK    :FOR TEST
011360 MOV      WORK,RSBA   :LOAD ADDR
011366 MOV      #67,RSCSI  :DO FUNCTION 67
011374 15: TSTB      RSCSI :DONE YET?
011400 BPL      IS         :NO
011402 MOV      RSBA,BAD   :GET BA REG
011406 MOV      @OBUF SV,GOOD :GET CORRECT ANS FOR RSBA
011412 CMP      GOOD,BAD  :IS RSBA CORRECT?
011414 BEQ      .+4      :YES
011416 HLT      :BAD=RSBA GOOD=CORRECT ANS.
011420 MOV      UNUM,GOOD  :GET UNIT NUMBER
011424 BIS      #1300,GOOD :SET IR AND OR BITS
011430 MOV      RSCS2,BAD  :GET CS2
011434 CMP      GOOD,BAD  :IS CS2 CORRECT?
011436 BEQ      .+4      :YES
011440 HLT      :BAD=CS2 GOOD=CORRECT ANS
011442 CMP      #1,RSER   :IS RSER CORRECT?
011450 BEQ      .+4      :YES
011452 HLT      :ER IS WRONG
011454 CLDK          :CLEAR ALL RS REG
011456 TST      RSER      :DID ERROR CLEAR
011462 BEQ      .+4      :YES
011464 HLT      :DS
011466 MOV      # -100,RSWC :SET WC TO -100
011474 MOV      @OBUF SV,WORK :GET OUTBUF ADD.
011502 ADD      #200,WORK   :FOR TEST
011510 MOV      WORK,RSBA   :LOAD ADDR
011516 MOV      #67,RSCSI  :DO FUNCTION 67
011524 25: TSTB      RSCSI :DONE YET?
011530 BPL      25        :NO

```

G05

```

011532 3S:  MOV 2RUF SV GOOD :GET CORRECT ANS.
011534  MOV 2R A BAD :GET BA REG
011542  CMP GOOD,BAD :IS RSBA CORRECT?
011544  BEQ +4 :YES
011546  HLT :BAD=RSBA GOOD=CORRECT ANS.
011550  MOV UNNUM,GOOD :GET UNIT NUMBER
011554  BIS #1300,GOOD :SET IR AND OR BITS
011560  MOV 2RSCS2 BAD :GET CS2
011564  CMP GOOD,BAD :IS CS2 CORRECT?
011566  BEQ +4 :YES
011570  HLT :BAD=CS2 GOOD=CORRECT ANS
011572  CMP #1,2R SER :IS RSER CORRECT?
011600  BEQ +4 :YES
011602  HLT :ER IS WRONG
011604  MOV #40011,2RSCS1 :CLEAR ERRORS
011612  CMP #4210,2RSCS1 :DID THEY CLEAR IN CS1
011620  BEQ +4 :YES
011622  HLT :NO
011624  TST 2R SER :DID RSER CLEAR
011630  BEQ +4 :YES
011632  HLT :NO
011634  TST 2R SAS :DID RSAS CLEAR
011640  BEQ +4 :YES
011642  HLT :NO

```

\*\*\*\*\*  
:TEST 63 TEST PAR IN RSER  
\*\*\*\*\*

```

011644 TST63: SCOPE
PARTST: MOV TIMSV,TIMES :RESTORE LOOP #
011654 CLDK :CLEAR ALL RS REG
1S: MOV #10,2R SER :SET PAR
011664 CMP #150600,2RSDS :DID ERR,ATA AND DRY SET?
011672 BEQ +4 :YES
011674 HLT :ER SHOULD SET IF PAR SETS IN RSER
011676 CLDK :CLEAR ALL RS REG
011700 TST 2R SER :DID PAR CLEAR?
011704 BEQ +4 :YES
011706 HLT :PAR DID NOT CLEAR BY CLEAR BIT
011710 CMP #10600,2RSDS :DID ERROR BITS CLEAR
011716 BEQ +4 :YES
011720 HLT :NO

```

H05

:CHECK BITS 12 TO 15 FOR 0  
:CHECK SECTOR FRACTION TO WATCH FOR MOVEMENT  
:CHECK CS BITS IN LA AND ADDRESS CONFIRM IN MR REG

:\*\*\*\*\*  
:TEST 64 LOOK AHEAD TEST  
:\*\*\*\*\*

011722 TST64: SCOPE

011724 LATST: BIT #17000,RSLSA :ARE BITS 12 TO 15 CLEARED?  
011732 BEQ +4 :YES  
011734 HLT !LA :BITS 12 TO 15 SHOULD BE CLEARED  
011736 TST65: SCOPE

:NOW TEST MOVEMENT IN SF BITS

011740 MOV #171005,WORK :SET UP FOR TIME OUT ROUTINE  
011746 MOV RSLSA,GOOD :GET READING FROM LA  
011752 BIC #7700,GOOD :GET RID OF CS BITS  
011756 1S: DEC WORK :WAIT FOR DISK  
011762 BEQ ERRR :TYPE ERROR  
011764 MOV RSLSA,BAD :READ LA  
011770 BIC #7700,BAD :CLEAR CS BITS  
011774 CMP GOOD,BAD :DID SF BITS CHANGE?  
011776 BEQ 1S :WAIT FOR TIME OUT  
012000 BR LATDON :LA OK CONT  
012002 ERRR: TYPE +2 :ASCIZ (15)(12)"SECTOR FRACTIONS NOT MOVING"  
012044 HLT !LA :TYPE LOOK AHEAD REG  
012046 LATDON: :DONE CONT.

I05

\*\*\*\*\*  
 :TEST 66 CHECK CS BITS TO INCREMENT AND ADDRESS CONFIRM BIT IN MR  
 \*\*\*\*\*

```

012046 TST66: SCOPE
012050 CSTST: MOV TIMES,TIMSV ;SAVE LOOP CT
012056 MOV #10,TIMES ;LOOP 10 TIMES
012064 CLDRK ;CLEAR ALL RS REG.
012066 MOV #1000,GOOD ;LOAD COUNTER
012072 BIT #BIT9,RSMR ;IS ADD CONFIRM BIT 0?
012100 BEQ ADDCF ;YES CONTINUE
012102 DEC GOOD ;WAIT FOR
012104 BNE -2 ;DISK TO MOVE
012106 BIT #BIT9,RSMR ;IS ADD. CON. BIT BIT 0?
012114 BEQ +4 ;YES
012116 HLT !MR ;ADD. CONF. BIT ALWAYS A 1
    
```

:NOW TEST TA BITS AND ADD. CON. BIT IN MR

```

012120 ADDCF: MOV #-1,RSDA ;INIT RSDA
012126 IS: MOV #-1,WORK ;SETUP TIMEOUT COUNTER
012134 INC RSDA ;GET NEXT SECTOR
012140 CMP #1000,RSDA ;DONE ALL YET?
012146 BEQ DONCS ;YES
012150 2S: DEC WORK ;DO TIMEOUT ROUTINE
012154 BEQ TMEOUT ;ADD. CON. NEVER CAME UP
012156 BIT #1000,RSMR ;DID ADD CONFIRM BIT SET?
012164 BEQ 2S ;YES
012166 MOV RSLSA,BAD ;GET LA
012172 BIC #77,BAD ;CLEAR SF BITS
012176 3S: MOV #6,WORK1 ;SET UP COUNTER
012204 ROR BAD ;MOV SA BITS RIGHT
012206 DEC WORK1 ;DO 6 TIMES
012212 BNE 3S ;DONE YET?
012214 MOV RSDA,GOOD ;GET DA
012220 BIC #177700,GOOD ;CLEAR JUNK
012224 CMP GOOD,BAD ;ARE SA BITS = IN DA AND LA REG.?
012226 BEQ +4 ;OK
012230 HLT ;GOOD =DA BAD = LA
012232 BR 1S ;NO WAIT

012234 TMEOUT: HLT !MR!ER!DS ;ADDRESS CONFIRM BIT NEVER SET COULD BE BAD OR
012236 DONCS: ;BAD LA OR BAD COMPARE BETWEEN LA AND DA
;TEST DONE CONTINUE
    
```

\*\*\*\*\*  
:TEST 67 PARITY TEST  
\*\*\*\*\*

012236 †TST67: SCOPE

```

012240 PART: MOV TMSV, TIMES ;RESTORE LOOP COUNTER
012246 MOV #PRTP, R2 ;SETUP TIME OUT VECTOR
012254 MOV #340, R2
012262 MOV #MPRO, R2 ;GET PAR REG
012266 TSTAGN: TST (R2) ;DOES IT EXIST
012270 MOV #WWP, R2 ;YES SET WRITE WRONG PARITY
012274 MOV #PARITY, R1 ;GET TEST LOCATION
012300 1S: MOV R1, R1 ;WRITE WRONG PARITY
012302 TST R1 ;READ IT
012304 TST R2 ;DID PARITY ERROR SET?
012306 BMI 2S ;YES
012310 CLR R2 ;CLEAR PARITY REG
012312 BR PRTP1 ;GET NEXT PARITY REG
012314 2S: BIC #100005, R2 ;TURN OFF WWP, ENABLE BAD PARITY, ACTION ENABLE
012320 TST R2 ;PARITY ERROR?
012322 BEQ NOPAR ;IF NO BR
012324 MOV #1, R2 ;CLEAR WWP AND ENABLE BAD PARITY AND ACTION ENABLE

```

```

012330 CLRDK ;CLEAR ALL RS REG
012332 MOV #PARITY, R2SBA ;SET UP CURRENT ADDRESS
012340 MOV #-1, R2SVC ;SET WORD COUNT TO -1
012346 MOV #61, R2SCS1 ;GO WRITE
012354 3S: TSTB R2SCS1 ;DONE YET?
012360 BPL 3S ;NO WAIT
012362 MOV R2SCS2, BAD ;GET CS2
012366 MOV #20100, GOOD
012372 BIS UNNUM, GOOD ;GET CORRECT AND FOR CS2
012376 CMP GOOD, BAD ;IS CS2 CORRECT?
012400 BEQ .+4 ;YES
012402 HLT ;CS2 SHOULD = GOOD
012404 CMP #144260, R2SCS1 ;IS CS1 CORRECT?
012412 BEQ .+4 ;YES
012414 HLT ;DS
012416 JMP NOPAR ;GET OUT

```

;TRAPOUT ROUTINE

```

012422 PRTP: CMP (6)+, (6)+ ;CLEAR STACK
012424 PRTP1: CMP #172136, R2 ;DONE YET?
012430 BEQ NOPAR ;YES NO PAR REG
012432 ADD #2, R2 ;NO TRY AGAIN
012436 JMP TSTAGN ;RETRY

```

```

012442 NOPAR: MOV #6, R2
012450 CLR R2
012454 MOV R1, R1 ;WRITE GOOD PARITY

```

\*\*\*\*\*  
:TEST 70 TEST WRITE CHECK ERROR  
\*\*\*\*\*

012456 †TST70: SCOPE

K05

:WRITE A WORD OF 0 AND FLOAT A 1 THROUGH IT TO CAUSE WCE  
 :SET BIT14 IN ONCEE AND WRITE A WD OF -1 AND FLOAT 0  
 :TO CAUSE WCE

012460	WCETST	CLROK	:CLEAR ALL RS REG
012462		CLR	:WRITE A WD OF 0
012466	WCETT:	MOV	:SET UP CURRENT ADDRESS
012474		MOV	:SET WORD COUNT TO -1
012502		MOV	:GO WRITE
012510	3\$:	TSTB	:DONE YET?
012514		BPL	:NO WAIT
012516		BIT	:WRITE A 1 OR 0?
012524		BEQ	:WRITE A 0
012526		MOV	:WRITE A 1
012534		CLC	:CLEAR CARRY
012536	6\$:	ROL	:FLOAT A 0 THROUGH BAD WD
012542		BCC	:DONE GET OUT
012544		BR	:CHECK WCE
012546	2\$:	CLR	:WRITE A 0
012552		SEC	:SET CARRY
012554	1\$:	ROL	:FLOAT A 1
012560		BCS	:GET OUT WHEN DONE
012562	5\$:	MOV	:SET UP CURRENT ADDRESS
012570		MOV	:SET WORD COUNT TO -1
012576		CLR	
012602		MOV	:GO WRITE CHECK
012610	4\$:	TSTB	:READY YET?
012614		BPL	:NO WAIT
012616		MOV	:GET CS2
012622		MOV	:SET UNIT #
012626		BIS	:SET BITS
012632		CMP	:IS CS2 CORRECT?
012634		BEQ	:YES
012636		HLT	:BAD=CS2 GOOD=CORRECT ANS
012640		MOV	:GET BAD WD THAT SHOULD CAUSE WCE
012644		CLR	:GET GOOD WD IF WRITING 0
012646		BIT	:ARE WE WRITING 1 OR 0
012654		BEQ	:0
012656		MOV	:GET GOOD WD FOR 1
012662	8\$:	HLT	:GOOD = CORRECT WD WRITTEN
			:BAD = INCORRECT WD THAT WCE DID NOT CATCH

L05

```

012664 7S:  CMP      #144250,2RSCS1  :DID TRE SET?
012672    BEQ      .+4              :YES
012674    HLT      !CS1!ER!DS      :TRE SHOULD SET IF WCE SETS
012676    MOV      2RSBA,BAD        :FETCH CURRENT ADDRESS
012702    MOV      2#0BUF SV,GOOD  :WHAT RSBA SHOULD EQUAL
012706    ADD      #2,GOOD          :UPDATE IT
012712    CMP      BAD,GOOD        :IS RSBA CORRECT
012714    BEQ      .+4              :YES EXECUTE CONTINUE
012716    HLT      !CS1!ER!DS      :RSBA FAILED TO INCREMENT
012720    CLRDK   UNNUM,GOOD       :CLEAR ALL RS REG
012722    MOV      #100,GOOD      :PUT DRIVE IN GOOD
012726    BIS      2RSCS2,BAD      :SET IR BIT
012732    MOV      GOOD,BAD       :GET CS2
012736    CMP      GOOD,BAD       :IS CS2 CORRECT
012740    BEQ      .+4              :YES
012742    HLT      !CS1!ER!DS      :BAD =CS2 GOOD IS CORRECT ANS
012744    CMP      #4200,2RSCS1  :DID TRE CLEAR?
012752    BEQ      .+4              :YES
012754    HLT      !CS1!ER!DS      :TRE DID NOT CLEAR WITH CLEAR
012756    BIT      #BIT14,ONCEE  :FLOATION A 1 OR 0?
012764    BEQ      1$            :FLOAT 1
012766    BR      6$              :FLOAT 0
012770    WCEDNE: BIS      #BIT14,ONCEE :SET BIT14
012776    CLRDK   UNNUM,GOOD       :CLEAR TEST FLAG
013000    MOV      #-1,OUTBUFF     :NOW WRITE -1 IN OUTBUF
013006    JMP      WCE↑          :CLEAR TEST FLAG
013012    WCEDON: BIC      #BIT14,ONCEE
    
```

M05

MAINDEC-11-DZRSB-F  
DZRSBF.P11

RH11-RS03LA-RS03-RS04  
27-SEP-77 15:12

BASIC FUNCTION DIAGNOSTIC MACY11 30(1046)  
TST71 TEST WRITE CHECK ERROR ON -B- PORT

28-SEP-77 10:04 PAGE 96

SEQ 0063

\*\*\*\*\*  
:TEST 71 TEST WRITE CHECK ERROR ON -B- PORT  
\*\*\*\*\*

013020 TST71: SCOPE

:WRITE A WORD OF 0 AND FLOAT A 1 THROUGH IT TO CAUSE WCE  
:SET BIT14 IN ONCEE AND WRITE A WD OF -1 AND FLOAT 0  
:TO CAUSE WCE

013022	WCETSB:	BIT	#BIT13,ONCEE	:-B- PORT?
013030		BEQ	1\$	:YES
013032		JMP	WCEDOS	:NO GET OUT
013036	1\$:	MOV	BPORTT,OBUSV	:GET -B- PORT BUFFER
013044		CLRDK		:CLEAR ALL RS REG
013046		CLR	BPORTT	:WRITE A WD OF 0
013052	WCETB:	MOV	B#OBUSV,RSBA	:SET UP CURRENT ADDRESS
013060		MOV	#-1,RSWC	:SET WORD COUNT TO -1
013066		MOV	#2061,RS1	:GO WRITE
013074	3\$:	TSTB	RS1	:DONE YET?
013100		BPL	3\$	:NO WAIT
013102		BIT	#BIT14,ONCEE	:WRITE A 1 OR 0?
013110		BEQ	2\$	:WRITE A 0
013112		MOV	#-1,BPORTT	:WRITE A 1
013120		CLC		:CLEAR CARRY
013122	6\$:	ROL	OUTBUF	:FLOAT A 0 THROUGH BAD WD
013126		BCC	WCEDOS	:DONE GET OUT
013130		BR	5\$	:CHECK WCE
013132	2\$:	CLR	BPORTT	:WRITE A 0
013136		SEC		:SET CARRY
013140	1\$:	ROL	BPORTT	:FLOAT A 1
013144		BCS	WCEDNB	:GET OUT WHEN DONE
013146	5\$:	MOV	B#OBUSV,RSBA	:SET UP CURRENT ADDRESS
013154		MOV	#-1,RSWC	:SET WORD COUNT TO -1
013162		CLR	RSDA	
013166		MOV	#2051,RS1	:GO WRITE CHECK
013174	4\$:	TSTB	RS1	:READY YET?
013200		BPL	4\$	:NO WAIT
013202		MOV	RS2,BAD	:GET CS2
013206		MOV	UNNUM,GOOD	:SET UNIT #
013212		BIS	#40300,GOOD	:SET BITS
013216		CMP	GOOD,BAD	:IS CS2 CORRECT?
013220		BEQ	7\$	:YES
013222		HLT		:CS2=BAD GOOD=CORRECT ANS
013224		MOV	BPORTT,BAD	:GET BAD WD THAT SHOULD CAUSE WCE
013230		CLR	GOOD	:GET GOOD WD IF WRITING 0
013232		BIT	#BIT14,ONCEE	:ARE WE WRITING 1 OR 0
013240		BEQ	8\$	:0
013242		MOV	#-1,GOOD	:GET GOOD WD FOR 1
013246	8\$:	HLT		:GOOD = CORRECT WD WRITTEN
				:BAD = INCORRECT WD THAT WCE DID NOT CATCH

N05

```

013250 7S:   CMP      #146250, @RSCS1  :DID TRE SET?
013256     BEQ      +4          :YES
013260     HLT      :CS1!ER!DS  :TRE SHOULD SET IF WCE SETS
013262     MOV      @RSBA, BAD  :FETCH CURRENT ADDRESS
013266     MOV      @#OBUFSV, GOOD :WHAT RSBA SHOULD EQUAL
013272     ADD      #2, GOOD     :UPDATE IT
013276     CMP      BAD, GOOD   :IS RSBA CORRECT
013300     BEQ      +4          :YES EXECUTE CONTINUE
013302     HLT      :RSBA FAILED TO INCREMENT
013304     CLDRK    :CLEAR ALL RS REG
013306     MOV      UNNUM, GOOD  :PUT DRIVE IN GOOD
013312     BIS      #100, GOOD   :SET IR BIT
013316     MOV      @RSCS2, BAD  :GET CS2
013322     CMP      GOOD, BAD   :IS CS2 CORRECT
013324     BEQ      +4          :YES
013326     HLT      :BAD =CS2 GOOD IS CORRECT ANS
013330     CMP      #4200, @RSCS1 :DID TRE CLEAR?
013336     BEQ      +4          :YES
013340     HLT      :CS1      :TRE DID NOT CLEAR WITH CLEAR
013342     BIT      #BIT14, ONCEE :FLOATION A 1 OR 0?
013350     BEQ      1$         :FLOAT 1
013352     BR      6$         :FLOAT 0
013354 WCEDNB: BIS      #BIT14, ONCEE :SET BIT14
013362     CLDRK    :
013364     MOV      #-1, @BPORTT  :
013372     JMP      WCE1B      :NOW WRITE -1 IN OUTBUF
013376 WCEDOS: MOV      @OUTBUF, @OBUFSV :RESTORE @OBUFSV
013404     BIC      #BIT14, ONCEE :CLEAR TEST FLAG
  
```

\*\*\*\*\*  
:TEST 72 TEST PROGRAM ERROR BIT IN RSCS2  
\*\*\*\*\*

013412 TST72: SCOPE

013414	PGETST: CLRDK		: CLEAR ALL RS REG
013416	MOV	#177777,OUTBUF	: DATA TO BE X-FERED
013418	MOV	#200LFSV,RSBDR	: SET UP CURRENT ADDRESS
013420	MOV	#177000,RSWC	: SET WORD COUNT
013422	MOV	#51,RS1	: GO WRITE
013424	25: TSTB	RS1	: IS ROY CLEARED YET?
013426	BMI	RS	: NO WAIT
013428	MOV	#71,RS1	: GO READ
013430	JSR	PC,INTRY	: WAIT FOR READY
013432	HLT	CS1	: ROY NEVER CAME UP
013434	CTD	#144260,RS1	: IS CS1 CORRECT?
013436	BEO	:+4	: YES
013438	HLT	CS1	: TRE SHOULD SET BY SETTING PGE
013440	TST	RSAS	: RS SHOULD = 0
013442	BEO	:+4	: YES
013444	HLT	RS	: RSAS SHOULD = 0
013446	MOV	UNUM,GOOD	: GET UNIT #
013448	BIS	#2300,GOOD	: SET PGE, IR, AND OR
013450	MOV	RS1,RS1	: GET CS1
013452	CTD	GOOD,BAD	: IS IT CORRECT?
013454	BEO	:+4	: YES
013456	HLT	RSWC	: BAC = CS2
013458	BNE	:+4	: SHOULD NOT BE 0
013460	HLT	MC:CS1	: BECAUSE PGE SHOULD ABORT
013462	TST	RSER	: CURRENT OPERATION
013464	BEO	:+4	: DID ANY ERRORS SET?
013466	HLT	DS	: NO
013468	BIS	TRE,RS1	: RMR SHOULD BE SET
013470	BIC	PC,GOOD	: CLEAR ERRORS
013472	MOV	RS1,RS1	: CLEAR PGE ERROR
013474	CTD	GOOD,BAD	: GET CS2
013476	BEO	:+4	: IS CS2 CORRECT?
013478	HLT	CS2	: YES
013480	CTD	#4260,RS1	: PGE DID NOT CLEAR BY CLEARING TRE BAC = CS2
013482	BEO	:+4	: DID SC CLEAR
013484	HLT	DS	: YES
013486			: DID NOT CLEAR BY CLEARING TRE

\*\*\*\*\*  
:TEST 73 TEST RMR IN RSR REGISTER TRYING TO WRITE INTO RSDA  
\*\*\*\*\*

013612 TST73: SCOPE

013614	RMRT1:	CLRDK		: CLEAR ALL RS REG
013616		MOV	280B0F5V, RRSBA	: SET UP CURRENT ADDRESS
013618		MOV	0177700, RRSWC	: SET WORD COUNT
013620		MOV	061, RRS1	: GO WRITE
013622	IS:	MOV	28RSCS1	: IS RDY SET?
013624		BIC	28	: YES WAIT FOR IT TO CLEAR
013626		CHP		: CAUSE ERROR
013628		BNE		
013630		MOV	SI	
013632		MOV	SDA	
013634		JSR	PC	
013636		HLT	!	
013638		CHP	!	
013640		BEG	!	
013642		HLT	!ER	
013644		CHP	07777, RRSDA	
013646		BNE	+4	
013648		HLT	!DA	
013650		CHP	0150600, RRSOS	
013652		BEG	+4	
013654		HLT	!DS!ER	
013656		CHP	0104230, RRS1	
013658		BEG	+4	
013660		HLT	!DS	
013662		CLRDK		
013664		TST	RRSER	
013666		BNE	+4	
013668		HLT	!ER	
013670		CHP	04200, RRS1	
013672		BEG	+4	
013674		HLT	!DS	

\*\*\*\*\*  
:TEST 74 TEST RMR IN RSR REGISTER TRYING TO WRITE INTO RSR  
\*\*\*\*\*

013770 TST74: SCOPE

013772	RMRT2:	CLRDK		: CLEAR ALL RS REG
013774		MOV	280B0F5V, RRSBA	: SET UP CURRENT ADDRESS
014002		MOV	0177700, RRSWC	: SET WORD COUNT
014010		MOV	061, RRS1	: GO WRITE
014016	28:	TSTB	RRS1	: IS RDY SET?
014022		BMI	28	: YES WAIT FOR IT TO CLEAR
014024		MOV	0177773, RRSER	: CAUSE ERROR

```

014032 JSR PC WAITRY
014036 HLT !CSI :ROY NEVER CAME UP
014040 CMP #4,RSER :DID RMR SET?
014044 BEQ +4 :YES
014048 HLT !ER :ER SHOULD = 4
014052 CMP #150600,RSOS :DID ERR SET?
014056 BEQ +4 :YES
014060 HLT !DS!ER :ERR DID NOT SET BECAUSE OF RMR
014064 CMP #144260,RSOS1 :IS CSI CORRECT?
014068 BEQ +4 :YES
014072 HLT !DS :CSI SHOULD = 144260
014076 CLRK :CLEAR ALL RS REG

```

```

:*****
:TEST 75 TEST RMR IN RSR REGISTER TRYING TO WRITE INTO RSCS1
:*****

```

```

014100 TST75: SCOPE
014102 RMRT3: CLRK :CLEAR ALL RS REG
014104 MOV #80B1F5V,RSBA :SET UP CURRENT ADDRESS
014108 MOV #177700,RSWC :SET WORD COUNT
014112 MOV #61,RSOS1 :GO WRITE
014116 ZS: TSTB RSOS1 :IS ROY SET?
014120 BHI ZS :YES WAIT FOR IT TO CLEAR
014124 MOV #30,RSOS1 :CAUSE ERROR
014128 JSR PC WAITRY :WAIT FOR READY
014132 HLT !CSI :ROY NEVER CAME UP
014136 CMP #4,RSER :DID RMR SET?
014140 BEQ +4 :YES
014144 HLT !ER :ER SHOULD = 4
014148 CMP #150600,RSOS :DID ERR SET?
014152 BEQ +4 :YES
014156 HLT !DS!ER :ERR NOT SET BECAUSE OF RMR
014160 CMP #144260,RSOS1 :IS CSI CORRECT?
014164 BEQ +4 :YES
014168 HLT !DS :CSI SHOULD = 144260
014172 CLRK :CLEAR ALL RS REG

```

E06

\*\*\*\*\*  
:TEST 76 TEST THAT RMR DOES NOT SET BY WRITTING INTO RSAS  
\*\*\*\*\*

014210 TST76: SCOPE

014212	RMR4:	CLROK		: CLEAR ALL RS REG
014214		MOV	#080BFSV,RSBA	: SET UP CURRENT ADDRESS
014222		MOV	#177700,RSWC	: SET WORD COUNT
014230		MOV	#172060,R3	: GET RSLA REG
014234	15:	MOV	#3,R4	: WAIT FOR
014236		BIC	#17,R4	: THE MIDDLE
014242		CMP	#20,R4	: OF SECTOR 0
014246		BNE	15	: BEFORE DOING A SEARCH
014250		MOV	#31,RS1	: SEARCH
014256		MOV	#0,RSAS	: TRY TO CAUSE ERROR
014264		CLR	WORK	: CLEAR COUNTER
014270	25:	BIT	#BIT7,RS1	: WAIT FOR DRY
014276		BNE	35	: READY CONT
014300		INC	WORK	: COUNT
014304		BNE	25	: RETRY
014306		HLT	:CS1	: RMY NEVER CAME UP
014310	35:	TST	RSER	: DID RMR SET?
014314		BEO	:+4	: NO
014316		HLT	:ER	: ER SHOULD = 0
014320		CMP	#110600,RS1	: DID ERR SET?
014326		BEO	:+4	: NO
014330		HLT	:DS:ER	: DS SHOULD = 110600
014332		CMP	#104230,RS1	: IS CS1 CORRECT?
014340		BEO	:+4	: YES
014344		HLT	:DS	: CS1 SHOULD = 144260
014346		CLROK		: CLEAR ALL RS REG
014354		CMP	#4200,RS1	: IS CS1 CORRECT?
014356		BEO	:+4	: YES
		HLT	:DS	: NO

\*\*\*\*\*  
:TEST 77 TEST DCK IN RSER  
\*\*\*\*\*

014360 TST77: SCOPE

:DO A WRITE AND THEN A CLEAR FUNCTION THAT SHOULD CAUSE DCK TO SET

```

014362 DCKTST: CLRDK          :CLEAR ALL RS REG
014364      CMP          #4,RS04DT  :IS THIS A LA DISK?
014372      BNE          7$          :NO
014374      MOV          #177640,WORK2 :GET WC FOR LA DISK
014402      BR           1$          :CONTINUE
014404 7$:      MOV          #177500,WORK2 :GET WC FOR RS04
014412      TST          RS04DT      :IS THIS A RS04?
014416      BNE          1$          :YES
014420      MOV          #177600,WORK2 :NO
014426 1$:      MOV          WORK2,RSWC :LOAD WC
014434      MOV          #-1,OUTBUF   :WRITE -1
014442      MOV          #0808,RSBA  :SET UP CURRENT ADDRESS
014450 4$:      BIS          #10,RS0C2 :SET BAI BIT
014456      MOV          #61,RS0C1   :GO WRITE
014464 5$:      TSTB         RS0C1    :IS RDY SET?
014470      BPL          5$          :WAIT FOR WRITE TO FINISH
014476 2$:      CLR          RS0DA   :SET DSK ADDRESS TO 0
014484      CLR          OUTBUF     :WRITE 0
014500      MOV          #0808,RSBA  :SET UP CURRENT ADDRESS
014510      MOV          #-1,RSWC    :LOAD WC
014516      MOV          #172060,R2  :PUT RSLA ADDR INTO R2
014524 3$:      MOV          (R2),R3  :GET LA AND WAIT FOR
014530      BIC          #77,R3      :SECTOR 40
014534      CMP          #4000,R3    :BEFORE
014538      BNE          3$          :WRITING
014542      MOV          #61,RS0C1   :GO WRITE
014546 6$:      MOV          (R2),R3  :GET RSLA AND WAIT FOR
014552      BIC          #17,R3      :MIDDLE OF SECTOR
014556      CMP          #20,R3     :0 BEFORE EXECUTING
014560      BNE          6$          :A CLEAR FUNCTION
014566      MOV          #40,RS0C2   :CLEAR ALL REG. DO IT THIS WAY
                                :DO NOT USE TRAP

```

```

014574 INCH: TSTB 2RSCS1 : IS BUSY CLEARED
014600 BAI 5S : FLAG CLEARED
014602 HLT :CS1 : ROY NEVER CAME UP
014604 6S: MOV 280BUFSV,2RSBA : SET UP CURRENT ADDRESS
014612 MOV WORK2,2RSWC : LOAD WC
014620 MOV 871,2RSCS1 : GO READ
014626 5S: TSTB 2RSCS1 : IS RDY SET?
014632 BPL 5S : WAIT FOR READ TO FINISH
014634 CMP 810000,2RSER : DID DCK SET?
014642 BEQ +4 : YES
014644 HLT :ER : DCK DID NOT SET
014646 CMP 8150600,2RSDS : DID ERR SET?
014654 BEQ +4 : YES
014656 HLT :DS : ER DID NOT SET BY DCK
014660 CMP 8144270,2RSCS1 : IS CSI CORRECT?
014666 BEQ +4 : YES
014670 HLT :DS,DA
014672 MOV 2RSCS2,BAD : GET CS2
014676 MOV UNNUM,GOOD : GET UNIT #
014702 BIS 8100,GOOD : SET IR
014706 CMP GOOD,BAD : IS CS2 CORRECT?
014710 BEQ +4 : YES
014712 HLT
014714 MOV 8177700,GOOD : NO
014720 1S: MOV 2RSWC,BAD : DID TRANSFER STOP AT END OF SECTOR?
014724 CMP GOOD,BAD
014726 BEQ +4 : YES
014730 HLT : NO
014732 MOV 8OUTBUF,GOOD : GET BA
014736 CMP 84,RS04DT : LA DISK?
014744 BNE 7S : NO
014746 ADD 8100,GOOD : YES
014752 BR 3S : CONTINUE
014754 7S: TST RS04DT : RS04?
014760 BNE 2S : YES
014762 ADD 8200,GOOD : RS03
014766 BR 3S
014770 2S: ADD 8400,GOOD : GET CORRECT ANS FOR BA
014774 3S: MOV 2RSBA,BAD : GET BA
015000 CMP BAD,GOOD : IS BA CORRECT?
015002 BEQ +4 : YES
015004 HLT : NO
015006 CLRDK : CLEAR ALL RS REG
015010 TST 2RSER : DID DCK CLEAR?
015014 BEQ +4 : YES
015016 HLT :ER : DCK DID NOT CLEAR WITH CLEAR
015020 MOV 8177500,2RSWC : CLEAR DCK ON
015026 MOV 280BUFSV,2RSBA : DRIVE BY WRITING
015034 MOV 861,2RSCS1 : GOOD DATA
015042 4S: TSTB 2RSCS1 : ON DRIVE
015046 BPL 4S
    
```

: TEST THE ABILITY OF THE DISK CONTROL TO  
 : INCREMENT THE TRACK REGISTER.

: A ONE WORD WRITE WILL BE EXECUTED  
 : RSDA=7777 RSWC = -1  
 : AT THE COMPLETION OF THE WRITE RSDA = 10000  
 : \*\*\*\*\*  
 : TEST 100 TEST DISK ADDRESS REGISTER  
 : \*\*\*\*\*

015050 †TST100: SCOPE

015052	DKADR:	CLRDK	:	CLEAR ALL RS REG
015054		MOV	#177777, @RSWC	: SET WORD COUNT TO -1
015062		MOV	@#0BUFSV, @RSBA	: SET UP CURRENT ADDRESS
015070		MOV	#7777, @RSDA	: SET RSDA TO ALL ONES
015076		MOV	#61, @RSCS1	: GO WRITE ONE WORD
015104		JSR	PC WAITRY	: WAIT FOR READY
015110		HLT	!CS1	: RDY DID NOT COME UP
015112	SS:	CMP	@RSDA, #10000	: DOES RSDA=0
015120		BEG	+4	: RSDA OK
015122		HLT	!DA	: DA DID NOT INCREMENT

: \*\*\*\*\*  
 : TEST 101 TEST IAE ERROR  
 : \*\*\*\*\*

015124 †TST101: SCOPE

: IAE ERROR SHOULD SET ERR, ATA AND SC BITS

015126	IAERR:	CLRDK	:	CLEAR ALL RS REG
015130		MOV	#177777, @RSWC	: SET WC TO -1
015136		MOV	@#0BUFSV, @RSBA	: SET UP BUS ADDRESS
015144		MOV	#177777, @RSDA	: SET DA TO RECEIVE ERROR
015152		MOV	#61, @RSCS1	: GO WRITE ONE WD
015160	7S:	TSTB	@RSCS1	: TEST FOR ERR OR RDY
015164		BMI	+4	: OK CONT.
015166		BR	7S	: WAIT
015170		CMP	#2000, @RSER	: DID IAE SET?
015176		BEG	+4	: YES
015200		HLT	!ER	: IAE SHOULD BE SET
015202		CMP	#150600, @RSDS	: DID ERR SET?
015210		BEG	+4	: YES
015212		HLT	!DS!AS	: ERR SHOULD BE SET
015214		CMP	#144260, @RSCS1	: DID SC SET?
015222		BEG	+4	: YES
015224		HLT	!CS1	: SC SHOULD BE SET
015226		CLRDK		: CLEAR ALL RS REG
015230		TST	@RSER	: CLR ERRORS?
015234		BEG	+4	: YES
015236		HLT	!ER	: ERR DID NOT CLR WITH 40 IN CSE

: IN THIS ROUTINE THE PROGRAM WILL GENERATE A  
: NON-EXISTENT DISK ERROR

: \*\*\*\*\*  
: TEST 102 TEST FOR NON-EXISTENT DISK ERROR  
: \*\*\*\*\*

015240 TST102: SCOPE

015242	NEDTST: CLDK		: CLEAR ALL RS REG
015244	MOV	#401,WORK	: SET UP FOR N.E.D. NUMBER
015252	CLR	GOOD	: LOOK FOR NON-EXISTENT DRIVES
015254	15: BIT	WORK,UNITSV	: ON THE SYSTEM
015262	BEQ	35	: FOUND NON-EXISTENT DRIVE
015264	INC	GOOD	: CONTAINS UNIT #
015266	ROL	WORK	: KEEP LOOKING
015272	BCS	NEDDON	: COULD NOT FIND ANY NON-EXISTENT DRIVES
015274	BR	15	: LOOK FOR NED
015276	35: MOV	GOOD,RS2	: LOAD NED IN CS2
015302	CLR	RS0A	: WRITE DRIVE REG
015306	TST	RSER	: DID ANY BITS SET IN RSER?
015312	BEQ	.+4	: NO
015314	HLT	:DS	: WHY DID RSER CHANGE?
015316	MOV	RS2,BAD	: GET CS2
015322	BIS	#10100,GOOD	: SET NED AND IR
015326	CMP	GOOD,BAD	: IS CS2 CORRECT?
015330	BEQ	.+4	: YES
015332	HLT		: GOOD=CORRECT CS2 BAD=CS2
015334	CMP	#160200,RS1	: IS CS1 CORRECT?
015342	BEQ	.+4	: YES
015344	HLT	:CS2	: TRE SHOULD SET BY NED ERROR
015346	TST	RSAS	: DID ANY BITS SET?
015352	BEQ	.+4	: NO
015354	HLT	:AS	: WHY DID AT BITS SET?
015356	MOVB	#100,RS1B	: CLEAR TRE
015364	BIT	#NED,RS2	: DID NED CLEAR
015372	BEQ	.+4	: YES
015374	HLT	:CS2	: NED DID NOT CLEAR
015376	MOV	RS0A,WORK	: READ DRIVE REG
015404	BIT	#NED,RS2	: DID NED SET?
015412	BNE	.+4	
015414	HLT	:DS	: NED DID NOT SET
015416	BR	NNDD	: GET OUT
015420	NEDDON: BIT	#BIT12.ONCEE	: DWAS THIS TYPED BEFORE?
015426	BNE	NNDD	: YES
015430	TYPE	.+2	: ASCIZ (15 (12) "COULD NOT FIND A NON-EXISTENT DRIVE"
0:5502	NNDD: BIS	#BIT12.ONCEE	: SET TYPED FLAG

J06

```

:*****
:TEST 103          TEST THAT DAO IN RSER AND LBT IN RSDS DO SET
:*****
015510  TST103: SCOPE

015512  DAOTST: CLRDK          :CLEAR ALL RS REG
015514  CMP          #4,RS04DT :RS03LA?
015522  BNE          3$        :NO
015524  MOV          #-41,RSWC  :LOAD WORD COUNT
015532  BR           1$        :CONT
015534  3$: MOV          #-201,RSWC :LOAD WC FOR RS04
015542  TST          RS04DT    :IS THIS A RS04?
015546  BNE          1$        :YES
015550  MOV          #-101,RSWC  :NO
015556  1$: MCV          #7777,RSDA :SET RSDA=TO ALL ONES
015564  2$: MOV          #0BUF SV,RSBA :CURRENT ADDRESS=OUTBUF
015572  MOV          #61,RSCSI  :WRITE
015600  JSR          PC, WAITRY :WAIT FOR READY
015604  HLT          !CS1      :RDY DID NOT SET
015606  CMP          #1000,RSER  :DID DAO SET?
015614  BEQ          +4        :YES
015616  HLT          !ER       :DAO DID NOT SET
015620  CMP          #152600,RSDS :DID LBT SET?
015626  BEQ          +4        :YES
015630  HLT          !DS       :LBT DID NOT SET
015632  TST          RS04DT    :IS ERROR FLAG SET
015636  BMI          +4        :ERROR IS SET
015640  HLT          !CS1      :SC DID NOT SET
015642  CLRDK          :CLEAR ALL RS REG
015644  CMP          #10600,RSDS  :DID ATA +LBT CLEAR
015652  BEQ          +4        :YES
015654  HLT          !DS       :ATA DID NOT CLEAR BY CLR BIT
015656  TST          RSER       :DID DAO CLEAR?
015662  BEQ          +4        :YES
015664  HLT          !ER       :DAO DID NOT CLEAR WITH CLEAR
    
```

\*\*\*\*\*  
:TEST 104 TEST THAT LBT DOES SET AND DAO DOES NOT  
\*\*\*\*\*

015666 TST104: SCOPE

015670	DAOTT:	CLRDK		: CLEAR ALL RS REG
015672		CMP	#4,RS04DT	: RS03LA?
015700		BNE	3\$	: NO
015702		MOV	#-37,RSWC	: LOAD WORD COUNT
015710		BR	1\$	: CONT
015712	3\$:	MOV	#-177,RSWC	: LOAD WC FOR RS04
015720		TST	RS04DT	: IS THIS A RS04?
015724		BNE	1\$	: YES
015726		MOV	#-77,RSWC	: NO
015734	1\$:	MOV	#7777,RS0A	: SET RS0A=TO ALL ONES
015742	2\$:	MOV	#0BUF5V,RSBA	: CURRENT ADDRESS=OUTBUF
015750		MOV	#61,RS0C1	: WRITE
015756		JSR	PC,WAITRY	: WAIT FOR READY
015762		HLT	:CS1	: RDY DID NOT SET
015764		TST	RSER	: ANY ERRORS?
015770		BEQ	:+4	: NO
015772		HLT	:ER	: YES
015774		CMP	#12600,RS0S	: DID LBT SET?
016002		BEQ	:+4	: YES
016004		HLT	:DS	: LBT DID NOT SET
016006		TST	RS0C1	: IS ERROR FLAG SET
016012		BPL	:+4	: NO
016014		HLT	:CS1	: ERROR
016016		CLRDK		: CLEAR ALL RS REG
016020		CMP	#10600,RS0S	: DID LBT CLEAR
016026		BEQ	:+4	: YES
016030		HLT	:DS	: ATA DID NOT CLEAR BY CLR BIT

\*\*\*\*\*  
:TEST 105 EXECUTE FUNCTION WITH ERROR BITS SET  
\*\*\*\*\*

016032 ↑TST105: SCOPE

016034	ERTST:	CLDK		:CLEAR ALL RS REG
016036		MOV	#177017, @RSER	:LOAD ER
016044		MOV	@RSAS, BAD	:GET AS REG
016050		MOV	UNCMP, GOOD	:GET UNIT ATA BIT
016054		BIC	#177400, GOOD	:CLEAR JUNK
016060		CMP	GOOD, BAD	:IS AS REG CORRECT?
016062		BEQ	:+4	:YES
016064		HLT	:AS	:AS BIT SHOULD BE SET
016066		CMP	#104200, @RSCS1	:DID ERAS SET IN CS1?
016074		BEQ	:+4	:YES
016076		HLT	:DS	:CS1 SHOULD =104200
016100		MOV	UNCMP, @RSAS	:CLEAR ATA BIT
016106		TST	@RSAS	:DID IT CLEAR?
016112		BEQ	:+4	:YES
016114		HLT	:AS	:COULD NOT CLEAR AS BIT
				:BY LOADING A 1 INTO IT
016116		CMP	#4200, @RSCS1	:DID SC CLEAR BY
016124		BEQ	:+4	:CLEARING ATA
016126		HLT	:ER	:NO
016130		MOV	#177777, OUTBUF	:DATA TO BE XFERED
016136		MOV	@#0BUFSV, @RSBA	:SET UP CURRENT ADDRESS
016144		MOV	#-1, @RSWC	:LOAD WC WITH -1
016152		MOV	#71, @RSCS1	:DO READ FUNCTION
016160		BIT	#1, @RSCS1	:DID GO BIT CLEAR
016166		BEQ	:+4	:YES
016170		HLT	:CS1	:GO BIT SHOULD BE CLEARED
016172	IS:	TSTB	@RSCS1	:WAIT FOR READY
016176		BPL	IS	:WAIT
016200		CMP	#144270, @RSCS1	:DID ERAS CLEAR BY SETTING GO BIT?
016206		BEQ	:+4	:YES
016210		HLT	:ER	:NO

M06

016212	MOV	@RSCS2,BAD	:GET CS2
016216	MOV	#1100,GOOD	:GET CORRECT ANS
016222	BIS	UNNUM,GOOD	:GET UNIT #
016226	CMP	GOOD,BAD	:IS CS2 CORRECT?
016230	BEQ	.+4	:YES
016232	HLT		:GOOD = WHAT CS2 SHOULD =
016234	CMP	#150600,@RSDS	:DID ERR BITS SET?
016242	BEQ	.+4	:NO
016244	HLT	:DS	:ERR BIT SHOULD BE 1
016246	CMP	#-1,@RSWC	:DID WC MOVE?
016254	BEQ	.+4	:NO
016256	HLT	:WC	:WC SHOULD = 1777777
016260	TST	@RSDA	:DID DA MOV
016264	BEQ	.+4	:NO
016266	HLT	:DA	:DA SHOULD =0
016270	CMP	@#0BUFSV,@RSBA	:DID BA MOVE
016276	BEQ	.+4	:NO
016300	HLT	:BA	:BA MOVED
016302	BIT	@RNCMP,@RSAS	:AS SHOULD BE SET
016310	BNE	.+4	:IS IT?
016312	HLT	:AS	:NO
016314	CMP	#177017,@RSER	:DID ER CHANGE?
016322	BEQ	.+4	:NO
016324	HLT	:ER	:ER SHOULD NOT CHANGE

\*\*\*\*\*  
:TEST 106 PAT AND MCPE TEST  
\*\*\*\*\*

016326 †TST106: SCOPE

016330	PATST: CLRDK		:CLEAR ALL RS REG
016332	BIS	#BIT4, @RSCS2	:SET PAT
016340	TST	@RSMR	:READ DRIVE REG
016344	MOV	@RSCS2, BAD	:GET CS2
016350	MOV	#120, GOOD	:MOPE SHOULD
016354	BIS	UNNUM, GOOD	:NOT SET
016360	CMP	GOOD, BAD	:IS CS2 CORRECT?
016362	BEQ	.+4	:YES
016364	HLT		:BAD = CS2 GOOD = CORRECT ANS
016366	MOV	#10, @RSMR	:CAUSE PAR TO SET IN RSER
016374	CMP	#10, @RSER	:DID PAR SET?
016402	BEQ	.+4	:YES
016404	HLT	!AS!DS	
016406	MOV	@RSAS, BAD	:GET AS REG
016412	MOV	UNCMP, GOOD	:GET UNIT ATA BIT
016416	BIC	#177400, GOOD	:CLEAR JUNK
016422	CMP	GOOD, BAD	:IS AS REG CORRECT?
016424	BEQ	.+4	:YES
016426	HLT	!AS	:AS BIT SHOULD BE SET
016430	CMP	#104200, @RSCS1	:DID ERRS SET IN CS1?
016436	BEQ	.+4	:YES
016440	HLT	!DS	:CS1 SHOULD =104200
016442	CLRDK		:CLEAR RS REG
016444	CMP	#4200, @RSCS1	:IS CS1 CORRECT?
016452	BEQ	.+4	:CLEARING ATA
016454	HLT	!ER	:NO
016456	MOV	@RSCS2, BAD	:CHECK TO SEE
016462	MOV	UNNUM, GOOD	:IF PAT CLEARS
016466	BIS	#100, GOOD	
016472	CMP	GOOD, BAD	
016474	BEQ	.+4	
016476	HLT		:PAT DID NOT CLEAR

\*\*\*\*\*  
:TEST 107 SET PAT BIT AND LOAD FUNCTION  
\*\*\*\*\*  
TST107: SCOPE

016500	SETPAT: CLARK		:CLEAR ALL REG
016502	BIS	00A1,0RSCS2	:SET BAI
016504	MOV	0177777,OUTBUF	:DATA TO BE XFERED
016512	MOV	000BUFSV,0RSBA	:SET UP CURRENT ADDRESS
016520	MOV	0-1000,0RSMC	:LOAD MC WITH -1
016528	BIS	0014,0RSCS2	:SET PAT BIT
016536	MOV	071,0RSCS1	:DO READ FUNCTION
016544	18: TSTB	0RSCS1	:WAIT FOR READY
016552	BPL	18	:WAIT
016560	CMP	0144270,0RSCS1	:DID CS1 GET LOADED?
016568	BEG	:+4	:NO
016576	HLT	:DS	:IT SHOULD NOT
016584	CMP	000BUFSV,0RSBA	:DID BA MOVE?
016592	BEG	:+4	:NO
016600	HLT	:BA	:YES
016608	CMP	0-1000,0RSMC	:DID MC MOVE?
016616	BEG	:+4	:NO
016624	HLT	:MC	:YES WHY?

\*\*\*\*\*  
:TEST 110 DO FUNCTION THEN SET PAT BIT  
\*\*\*\*\*  
TST110: SCOPE

016614	FUNDO: CLARK		:CLEAR ALL REG
016616	BIS	00A1,0RSCS2	:SET BAI
016620	MOV	0177777,OUTBUF	:DATA TO BE XFERED
016624	MOV	000BUFSV,0RSBA	:SET UP CURRENT ADDRESS
016628	MOV	0-1000,0RSMC	:LOAD MC WITH -1
016632	38: MOV	071,0RSCS1	:DO A READ
016636	BMI	38	:WAIT FOR BUSY
016640	BIS	0014,0RSCS2	:SET PAT
016644	28: TSTB	0RSCS1	:WAIT FOR READY
016648	BPL	28	
016652	CMP	0144270,0RSCS1	:DID MCPE SET?
016656	BEG	:+4	:NO
016660	HLT	:ER	:YES
016664	MOV	0RSCS2,BAD	:GET CS2
016668	MOV	0730,GOOD	:GET CORRECT ANS
016672	BIS	UNNUM,GOOD	:GET UNIT #
016676	CMP	GOOD,BAD	:IS CS2 CORRECT?
016680	BEG	:+4	:YES
016684	HLT		:GOOD = WHAT CS2 SHOULD =

```

016730 C *P #10600, @RSDS :DID ERR BITS SET?
016736 BEQ :+4 :NO
016740 HLT :DS :ERR BIT SHOULD BE 1
016742 TST @RSEB :IS ER CLEAR?
016746 BEQ :+4 :YES
016750 HLT :DS!DA :NO ERRORS SHOULD BE SET
016752 CLROK :CLEAR ALL RS REG
016754 CMP #4200, @RSCS1 :IS CS1 CORRECT?
016762 BEQ :+4 :YES
016764 HLT :DS

```

```

:*****
:TEST 111 TEST PAR BY SETTING PAT
:*****

```

016766 \*ST111: SCOPE

```

016770 PATTST: CLROK :CLEAR ALL RS REG
016772 BIS #BA1, @RSCS2 :SET BAI
017000 MOV #177777, @OUTBUF :DATA TO BE XFERED
017006 MOV @R0BUFV, @R5BA :SET UP CURRENT ADDRESS
017014 MOV #1000, @R5WC :LOAD WC WITH -1
017022 MOV #61, @RSCS1 :DO WRITE FUNCTION
017030 18: TSTB @RSCS1 :WAIT FOR READY
017034 BIT 18 :WAIT
017036 28: BIS #BIT4, @RSCS2 :SET PAT
017044 TSTB @RSCS1 :WAIT FOR READY
017050 BPL 28
017052 CMP #144260, @RSCS1 :DID MCPE SET?
017060 BEQ :+4 :NO
017062 HLT :ER :YES
017064 MOV @RSCS2, @BAD :GET CS2
017070 MOV #230, @GOOD :GET CORRECT ANS-- DO NOT CHECK IR - REASON 50 CYCLE
017074 BIS @UNUM, @GOOD :GET UNIT #
017100 BIC #BIT6, @BAD :CLEAR IR BIT FOR CS2 COMPARE
017104 CMP @GOOD, @BAD :IS CS2 CORRECT?
017106 BEQ :+4 :YES
017110 HLT :GOOD = WHAT CS2 SHOULD =
017112 CMP #150600, @RSDS :DID ERR BITS SET?
017120 BEQ :+4 :NO
017122 HLT :DS :ERR BIT SHOULD BE 1
017124 CMP #10, @RSEB :DID PAR SET?
017132 BEQ :+4 :YES
017134 HLT :DS!DA :NO
017136 CLROK :CLEAR ALL RS REG
017140 CMP #4200, @RSCS1 :IS CS1 CORRECT?
017146 BEQ :+4 :YES
017150 HLT :DS
017152 TST @RSEB :DID PAR CLEAR?
017156 BEQ :+4 :YES
017160 HLT :DS

```

\*\*\*\*\*  
:TEST 112 TEST THE ABILITY TO FILL THE LAST SECTOR  
\*\*\*\*\*

017162 †TST112: SCOPE

017164	LASTSC: CLRDK		:CLEAR ALL RS REG
017166	MOV	#7777,RSDA	:SET RSDA=TO ALL ONES
017174	CMP	#4,RS04DT	:LA DISK?
017202	BNE	ZS	:NO
017204	MOV	#-40,RSWC	:LOAD WORD COUNT
017212	BR	IS	:CONTINUE
017214	2S: MOV	#-100,RSWC	:WORD COUNT=-100
017222	TST	RS04DT	:IS THIS A RS04?
017226	BEQ	IS	:NO
017230	MOV	#-200,RSWC	:YES
017236	1S: MOV	#0BUF5V,RSBA	:CURRENT ADDRESS=OUTBUF
017244	MOV	#61,RS0CS1	:WRITE
017252	JSR	PC,WAITRY	:WAIT FOR READY
017256	HLT	:CS1	:RDY DID NOT SET
017260	TST	RSER	:DID ANY ERROR BITS SET?
017264	BEQ	:+4	:NO
017266	HLT	:ER	:GOT AN ERROR
017270	CMP	#12600,RS0S	:DID LBT SET?
017276	BEQ	:+4	:YES
017300	HLT	:DS	:LBT DID NOT SET
017302	TST	RS0CS1	:IS ERROR FLAG SET
017306	BPL	:+4	:ERROR IS SET
017310	HLT	:CS1	:SC DID NOT SET
017312	CLRDK		:CLEAR ALL RS REG
017314	CMP	#10600,RS0S	:DID ATA +LBT CLEAR
017322	BEQ	:+4	:YES
017324	HLT	:DS	:ATA DID NOT CLEAR BY CLR BIT

:FILL SECTOR WITH ALL ONES.  
:NOW WRITE 1ST WORD IN SECTOR  
:TEST REMAINING 63 WORDS FOR 0

\*\*\*\*\*  
:TEST 113 TEST FOR ZERO'S IN PARTIAL FILLED SECTOR  
\*\*\*\*\*

```

017326 TST113: SCOPE
017330 SECT: CLROK
017332 MOV # -1,OUTBUF
017334 MOV @#0BUF SV,RSBA
017336 CMP #4,RS04D
017338 BNE 4$
017340 MOV # -40,RSWC
017342 BR 5$
017344 4$: MOV # -200,RSWC
017346 TST RS04D
017348 BNE 5$
017350 MOV # -100,RSWC
017352 5$: BIS #BAI,RS0CS2
017354 MOV #61,RS0CS1
017356 3$: TSTB RS0CS1
017358 BPL 3$
017360 CLR RS0DA
017362 MOV # -1,OUTBUF
017364 MOV @#0BUF SV,RSBA
017366 MOV # -1,RSWC
017368 BIS #10,RS0CS2
017370 MOV #61,RS0CS1
017372 1$: TSTB RS0CS1
017374 BPL 1$
017376 BIS #10,RS0CS2
017378 TST RS04D
017380 BEQ 7$
017382 MOV #200,WORK
017384 BR 8$
017386 7$: MOV #100,WORK
017388 8$: MOV @#0BUF SV,R1
017390 MOV # -1,(R1)+
017392 6$: CLR (R1)+
017394 DEC WORK
017396 BNE 6$
017398 CLR RS0DA
017400 MOV @#0BUF SV,RSBA
017402 CMP #4,RS04D
017404 BNE 11$
017406 MOV # -40,RSWC
017408 BR 10$
017410 11$: TST RS04D
017412 BEQ 9$
017414 MOV # -200,RSWC
017416 BR 10$
017418 9$: MOV # -100,RSWC
017420 10$: MOV #51,RS0CS1

```

```

: CLEAR ALL RS REG
: PUT - INTO OUTBUF
: SET UP CURRENT ADDR
: RS03LA DISK?
: NO
: LOAD WORD COUNT
: CONTINUE
: LOAD WC FOR RS04
: RS04?
: YES
: SET WORD COUNT TO -100
: SET BAI BIT
: WRITE
: IS RDY SET?
: NO
: SET DSK ADDRESS TO 0
: PUT 177777 INTO OUTBUF
: SET UP CURRENT ADDR
: SET WORD COUNT TO -1
: SET BAI BIT
: WRITE
: IS RDY SET?
: NO
: CLEAR BAI BIT
: RS04?
: NO
: YES
: CONT
: SET UP BUFFER
: GET STARTING ADDR OF BUF
: LOAD FIRST WD WITH -1
: LOAD REST WITH 0
: DONE YET?
: NO
: SET DSK ADDRESS TO 0
: SET UP CURRENT ADDR
: RS03LA DISK?
: NO
: LOAD WORD COUNT
: CONTINUE
: RS04?
: NO
: YES
: CONT
: SET WORD COUNT TO -100
: WRITE CHECK

```

```

017640 28: BIT      #200,2RSCS1      : IS RDY SET?
017646     BEQ      28             : NO
017650     MOV      UNNUM,GOOD      : GET UNIT #
017654     BIS      #100,GOOD      : SET IR BIT
017660     MOV      2RSCS2,BAD      : GET CS2
017664     CMP      GOOD,BAD        : IS CS2 CORRECT?
017666     BEQ      +4              : YES
017670     HLT      !ER             : THERE WAS A WRITE CHECK ERROR
    
```

\*\*\*\*\*  
 :TEST 114 IF MEMORY MANAGEMENT IS AVAILABLE CHECK THE EXTENDED MEMORY ADDRESS BITS  
 \*\*\*\*\*

017672 TST114: SCOPE

```

017674 EXTTST: CLRDK      : CLEAR ALL RS REG.
017676     MOV      TIMES,TIMSV    : SAVE LOOP #
017704     MOV      #10,TIMES       : LOOP 10 TIMES
017712     MOV      #EXTTRP,4     : SETUP TIMEOUT TRAP
017720     MOV      #340,6
017726     TST      2#SR0         : IF MEMORY MANAGEMENT IS NOT
                                : AVAILABLE THE PROGRAM WILL TRAP
                                : AND TRANSFER TO END OF THE TEST

017732     MOV      #EXTTRP,4
017740     MOV      #7600,2#KIPAR7 : OPEN I/O REGISTERS
017746     CLR      2#KIPAR0      : FREE FIRST 4K
017752     MOV      #200,2#KIPAR1 : ENABLE SECOND 4K
017760     MOV      #2000,2#KIPAR2
017766     MOV      #400*256.-400+UP+RW,2#KIPOR0 : SET KIPOR0=RW UP 400 BLOCKS
017774     MOV      #400*256.-400+UP+RW,2#KIPOR1 : SET KIPOR1=RW UP 400 BLOCKS
020002     MOV      #400*256.-400+UP+RW,2#KIPOR2 : SET KIPOR2=RW UP 400 BLOCKS
020010     MOV      #400*256.-400+UP+RW,2#KIPOR7 : SET KIPOR7=RW UP 400 BLOCKS
020016     MOV      #1,2#SR0      : TURN ON MEMORY MANAGEMENT
020024     MOV      #40000,R2     : R2 EQUALS BASE ADDR
    
```

G07

```

020030 7$: MOV      #177777,(R2)      :INSERT PATTERN INTO 200000
020034  MOV      #-2,RSWC          :SETUP WORDCOUNT
020042  MOV      #177777,RSBA        :SETUP BUS ADDR
020050  MOV      #61,RSCS1           :WRITE TWO WORDS ON DISK, RSBA
                                       :STARTS AT 177777 TO FORCE CARRY
                                       :TO SET A16
                                       :WAIT FOR READY

020056  TSTB     RSCS1              :
020062  BPL     .-4                 :
020064  TST     RSCS1              :
020070  BPL     1$                 :
020072  HLT     !ER!DA!DS         :STATUS ERROR AFTER 2 WORD WRITE
020074  BR     2$                 :USING MEXO
020076 1$: CMP     #4660,RSCS1     :IS CS1 CORRECT
020104  BEQ     3$                 :YES
020106  HLT     !ER              :CS2 DID NOT COMPARE
020110  BR     2$                 :
020112 3$: CLR     (R2)            :CLEAR LOCATION 200000
020114  CLR     RSDA              :SETUP DA
020120  MOV     #177777,RSBA      :SETUP BA
020126  MOV     #-2,RSWC          :SETUP WC
020134  MOV     #71,RSCS1         :READ TWO WORDS INTO LOCATIONS
                                       :177777 AND 200000.
                                       :WAIT FOR READY

020142  TSTB     RSCS1              :
020146  BPL     .-4                 :
020150  TST     RSCS1              :ANY ERRORS?
020154  BPL     4$                 :BRANCH IF NO
020156  HLT     !ER              :ERROR AFTER READING 2 WORDS
020160  BR     2$                 :
020162 4$: CMP     #4670,RSCS1     :IS CS1 CORRECT?
020170  BEQ     5$                 :YES
020172  HLT     !ER              :CS1 DID NOT COMPARE
020174  BR     2$                 :READ STARTING AT 177777
020176 5$: CMP     #177777,(R2)    :WAS DATA READ INTO LOCATION
020202  BEQ     2$                 :200000 CORRECTLY? - BRANCH IF YES
020204  MOV     #177777,GOOD      :
020210  MOV     (R2),BAD          :
020212  HLT     !ER              :DATA COMPARE ERROR AT 200000
020214  NOP
  
```

# H07

```

020216    EXTT1:    CLARK                                :CLEAR ALL REG
020220            MOV            #4000,2#KIPAR2                               
020226            MOV            #40000,R2                                :R2 EQUALS THE BASE ADDR
020232            75:    MOV            #177777,(R2)                                :INSERT PATTERN INTO 400000
020236            MOV            #177777,2RSBA                                :SETUP BUS ADDR
020244            MOV            #-2,2RSWC                                :LOAD WC
020252            MOV            #461,2RSCS1                                :SET BIT A16 AND WRITE
020260            TSTB            2RSCS1                                :WAIT FOR READY
020264            BPL            -4                               
020266            TST            2RSCS1                                :ANY ERRORS?
020272            BPL            45                                :BRANCH IF NO
020274            HLT            !ER                                :ERROR AFTER READING 2 WORDS
020276            45:    CMP            #5260,2RSCS1                                :IS CS1 CORRECT?
020304            BEQ            105                                :BRANCH IF YES
020306            HLT            !ER                                :NO CS1 DID NOT COMPARE
020310            105:    CLR            (R2)                                :READ STARTING AT 377777
020312                                :CLEAR LOCATION 400000
020320            MOV            #-2,2RSWC                                :READ TWO WORDS STARTING AT 377777
020324            CLR            2RSDA                                :SETUP WC
020328            MOV            #177777,2RSBA                                :SETUP DA
020332            MOV            #471,2RSCS1                               
020340            TSTB            2RSCS1                                :CLEAR A 17 SET A16,READ
020344            BPL            -4                                :WAIT FOR READY
020346            TST            2RSCS1                                :ANY ERRORS?
020352            BPL            115                                :BRANCH IF NO
020354            HLT            !ER                                :ERROR WHILE READING TWO WORDS
020356            BR            EXTRP                               
020360            115:    CMP            #5270,2RSCS1                                :IS CS1 CORRECT?
020366            BEQ            125                                :BRANCH IF YES
020370            HLT            !ER                                :CS1 DID NOT COMPARE
020372            125:    CMP            #177777,(R2)                                :READ STARTING AT 377777
020376            BEQ            EXTRP                                :WAS DATA READ INTO LOCATION 400000
020400            MOV            #177777,GOOD                                :CORRECTLY? - BRANCH IF YES
020404            MOV            (R2),BAD                               
020406            HLT                                :DATA COMPARE ERROR AT 400000 IF
020410            EXTRP:    CLR            2#SRO                                :RECEIVED=0 - LOCATION WASN'T ACCESSED
020414                                EXT1                                :TURN OFF MEMORY MANAGEMENT
020416            EXTTRP:    NOP                                :UPDATE TEST NUMBERS
020420            EXT1:    MOV            #500,SP                                :RESTORE STACK
020424            MEMOUT:    MOV            #6.4                               
020432                                CLR            6
  
```

\*\*\*\*\*  
 :TEST 115 TEST PROGRAM INTERRUPT BY MOVING 300 INTO RSCS1  
 :\*\*\*\*\*

020436 †TST115: SCOPE

```

020440 QES: CLRDK          :CLEAR ALL DRIVES
020442     MOV          #500,SP      :SETUP STACK
020446     MOV          #PGTRAP,RSVEC :SET UP VECTOR
020454     MOV          #340,RSVCPS  :SET TRAP PS
020462     MOV          #200,PS     :SET PS AT PRIORITY 4
020470     MOV          #300,RSCS1  :THIS SHOULD CAUSE A TRAP
020476     MOV          #500,WORK   :SETUP LOOP
020504 1$: DEC          WORK        :DEC LOOP SHOULD
020510     BNE          1$          :INTERUPE BEFORE LOOP IS DONE
020512     HLT          !CS1       :SHOULD NEVER GET HERE
020514     JMP          QESDON      :GET OUT

020520 PGTRAP: CMP        (6)+(6)+   :TRAP OK
020522     CMP          #4200,RSCS1  :DID IE CLEAR?
020530     BEQ          +4          :YES
020532     HLT          !CS1       :IE SHOULD BE CLEARED
020534 QESDON:
    
```

\*\*\*\*\*  
 :TEST 116 TEST THAT DISK DOES NOT INTERRUPT WHEN PS IS AT 5  
 :\*\*\*\*\*

020534 †TST116: SCOPE

```

020536 INTR5: MOV          #500,SP      :SETUP STACK
020542     MOV          TIMSV,TIMES  :RESTORE LOOP COUNTER
020550     CLRDK         :CLEAR ALL RS REG
020552     MOV          #INT112,RSVEC :SET UP INTERRUPT VECTOR
020560     MOV          #340,RSVCPS  :SET PRIO.
020566     MOV          #240,PS     :LOCK OUT ALL INTERRUPTS ABOVE
020574     MOV          PS,BAD      :GET PS
020600     MOV          #17777,RSWC  :SET WORD COUNT TO -1
020606     MOV          #0BUFSV,RSBA :LOAD CURRENT ADDRESS
020614     MOV          #161,RSCS1  :GO WRITE (INTERRUPT ENABLED)
020622     JSR          PC,WAITRY   :WAIT FOR READY
020626     HLT          !CS1       :NO RDY NEVER CAME UP
020630     BR          INTDON       :RESTART ROUTINE

:PROCESSOR SHOULD NOT TRAP TO INT112

020632 INT112: MOV        #240,GOOD   :WHAT PS SHOULD HAVE
020636     HLT         :GOOD = CORRECT ANS FOR PS
020640 INTDON:
    
```

```

:*****
:TEST 117 TEST THAT DISK DOES INTERRUPT WHEN PS IS AT 4
:*****
TST117: SCOPE
    
```

```

020640
020642 INTR4:  MOV    #500,SP           ;SETUP STACK
020646          CLRDK                ;CLEAR ALL RS REG
020650          MOV    #INT114, @RSVEC ;SET UP DISK TRAP VECTOR
020656          MOV    #340, @RSVCPS  ;SET PRIO.
020664          MOV    #200, @PS      ;SET PROCESSOR TO PRIORITY 4
020672          MOV    @PS, @BAD      ;GET PS
020676          MOV    #200, @GOOD    ;GET CORRECT PS
020702          MOV    #177777, @RSWC ;SET WORD COUNT TO -1
020710          MOV    @OBUFSV, @RSBA ;LOAD CURRENT ADDRESS
020716          MOV    #161, @RSCSI  ;WRITE (INTERRUPT ENABLE)
020724          CLR    WORK
020730          INC    WORK           ;WAIT FOR INTERRUPT TO OCCUR
020734          BNE    .-4
020736          HLT
020740          HLT    !ER!DS
020742          BR    DONINT         ;CONT
020744 INTR114: CMP    #4260, @RSCSI ;DID IE CLEAR?
020752          BEQ    +4            ;YES
020754          HLT    !CS1         ;WHY DID NOT IE CLEAR
020756 DONINT:
    
```

```

:*****
:TEST 120 TEST INTERRUPT ON ERROR
:*****
TST120: SCOPE
    
```

```

020756
020760 ERINT:  MOV    #500,SP           ;SETUP STACK
020764          MOV    #200, @PS      ;SET PS AT PRI 4
020772          MOV    #ERRINT, @RSVEC ;SET UP INTERRUPT ADD.
021000          CLRDK                ;CLEAR ALL RS REG
021002          MOV    #340, @RSVCPS  ;SET PRIO.
021010          MOV    #177777, @RSDA ;SET RSDA=TO ALL ONES
021016          MOV    #177600, @RSWC ;WORD COUNT=-200
021024          MOV    @OBUFSV, @RSBA ;CURRENT ADDRESS=OUTBUF
021032          MOV    #161, @RSCSI  ;WRITE
021040          JSR    PC, WAITRY    ;WAIT FOR READY
021044 IS:      HLT    !ER!DS        ;Y DIDN'T PGM INTERRUPT IS RDY SET?
021046          BR    FINTST        ;GET OUT
021050 ERRINT: CMP    #144260, @RSCSI ;IS CSI RIGHT?
021056          BEQ    +4            ;YES
021060          HLT    !ER!DS
021062          CMP    (6)+, (6)+    ;CLEAR STACK
021064 FINTST:
    
```

\*\*\*\*\*  
:TEST 121 DYNAMIC FUNCTION TEST  
\*\*\*\*\*  
↑TST121: SCOPE  
:EXECUTE FUNCTION MODIFY UNIT # AND DO A DRIVE SEARCH  
:DRIVE SEARCH WILL ONLY BE DONE IF THERE ARE AT LEAST 2 DRIVES  
:2ND DRIVE MAY NOT BE TESTED YET SO IF THIS TEST FAILS CHECK 2ND DRIVE  
:BEFORE TRYING TO DEBUG THIS TEST

021064  
021066 MODNUM: CLDK :CLEAR ALL RS REG  
021070 MOV TIMES,TIMSV :SAVE LOOP COUNT  
021076 MOV #10,TIMES :LOOP ONLY 10 TIMES  
021104 CLR WORK1 :CLEAR WORK LOC.  
021110 CLR R3  
021112 CLR R4  
021114 MOV #DVTAB,R2 :SETUP TABLE  
021120 MOV #401,WORK :SETUP TO TEST FOR MORE DRIVES  
021126 7S: BIT WORK,UNITSV :IS DRIVE ON SYSTEM?  
021134 BEQ 6S :NO  
021136 CMP R4,UNNUM :IS THIS THE SAME DRIVE?  
021142 BNE 8S :NO  
021144 6S: INC R4 :UPDATE 2ND UNIT #  
021146 CLC  
021150 ROL WORK :CHECK FOR NEXT DRIVE  
021154 BCC 7S :NOT DONE YET  
021156 BIT #BIT3,WORK1 :MULTI DRIVE?  
021164 BNE 12S :YES  
021166 MOV UNNUM,RS :LOAD UNIT NO  
021172 INC RS :CHANGE IT  
021174 BIC #177770,RS :CLEAR JUNK  
021200 BR 12S  
021202 8S: BIS #BIT3,WORK1 :SET FOUND MULTI DRIVE  
021210 MOV R4,(R2)+ :LOAD UNT # INTO TABLE  
021212 INC R3 :COUNT # OF DRIVES  
021214 MOV R3,SAVEE :SAVE IT  
021220 BR 6S  
021222 12S: MOV #TSTVEC,RSVEC :SETUP INT. TRAP  
021230 MOV #340,RSVCPS :SETUP PRIO.  
021236 CLR OUTBUF :CLR TO READ INTO  
021242 MOV #OBUFSV,RSBA :SET UP CURRENT ADDRESS  
021250 MOV #-1000,RSWC :SET WORD COUNT  
021256 1S: MOV #60,RSDA :LOAD DA  
021264 MOV #DVTAB,R2 :GET TABLE  
021270 MOV #161,RS1 :GO WRITE  
021276 TST R3 :MORE THEN 1 DRIVE?  
021300 BNE 13S :YES  
021302 MOV RS,RS2 :NO MODIFY UNIT #  
021306 BR 14S

```

021310 13$: MOV (R2)+, @RSCS2 ; LOAD UNIT#
021314 MOV #131, @RSCS1 ; DO SEARCH
021322 DEC R3 ; DONE ALL DRIVES YET?
021324 BNE 13$ ; NO
021326 14$: MOV #200, @#PS ; ENABLE INTERRUPTS
021334 WTDV: WAIT
021336 TSTVEC: MOV UNNUM, @RSCS2 ; GET 1ST DRIVE
021344 MOV @RSCS1, @BAD ; GET CS1
021350 BIC #BIT15, @BAD ; CLEAR SC
021354 MOV #4250, @GOOD ; GET CORRECT ANS
021360 CMP @GOOD, @BAD ; IS CS1 CORRECT?
021362 BEQ 4$ ; NO! X-FER OK
021364 HLT ; CS1 SHOULD = 14270 OR 4270
021366 HLT ;DS!AS
021370 4$: TST @RSCS2 ; TEST WC
021374 BEQ +4 ; WORD COUNT DID OVERFLOW
021376 HLT ;WC ; SHOULD = 0
021400 MOV UNNUM, @GOOD ; GET CORRECT
021404 BIS #100, @GOOD ; ANS OF CS2
021410 MOV @RSCS2, @BAD ; GET CS2
021414 CMP @GOOD, @BAD ; IS CS2 CORRECT?
021416 BEQ +4 ; YES
021420 HLT ;GOOD = CORRECT ANS FOR CS2
021422 MOV @RSCS2, @BAD ; FETCH CURRENT ADDRESS
021426 MOV @#OBUFSV, @GOOD ; WHAT RSBA SHOULD EQUAL
021432 ADD #2000, @GOOD ; UPDATE IT
021436 CMP @BAD, @GOOD ; IS RSBA CORRECT
021440 BEQ +4 ; YES EXECUTE CONTINUE
021442 HLT ;RSBA FAILED TO INCREMENT
021444 CMP #4, @RS04DT ; RS03LA?
021452 BNE 2$ ; NO
021454 CMP #100, @RSDA ; IS DA CORRECT?
021462 BEQ 3$ ; OK
021464 HLT ;DA!AS
021466 BR 3$ ; CONTINUE
021470 2$: TST @RS04DT ; IS THIS A RS04?
021474 BNE 5$ ; YES
021476 CMP #70, @RSDA ; IS DA CORRECT?
021504 BEQ 3$ ; YES
021506 HLT ;DA NOT CORRECT
021510 BR 3$ ; CONTINUE
021512 5$: CMP #64, @RSDA ; WAS RSDA INCREMENTED
021520 BEQ +4 ; RSDA OK
021522 HLT ;DA!ER!DS ; RSDA SHOULD CONTAIN A 64
021524 3$: MOV #TRE, @RSCS1 ; CLEAR ALL ERRORS IF ANY
021532 BIT #BIT3, @WORK1 ; MULTI DRIVE?
021540 BEQ WTDV1 ; NO
021542 1$: MOV #TTVEC, @RSVEC ; SETUP INT FOR NEXT DRIVE
021550 MOV #WTDV2, (SP) ; GET WAIT
021554 MOV #100, @RSCS1 ; SET IE
021562 RTI ; RETURN
021564 WTDV2: WAIT
    
```

```

;SERVICE ROUTINE FOR SEARCH FUNCTIONS
021566  ITVEC:  CLR      R2          ;CLEAR UNIT #
021570          CLC
021572          MOV      #401,WORK
021600  1$:   BIT      WORK,@RSAS  ;DID THIS DRIVE INT?
021606          BNE      2$
021610          INC      R2          ;UPDATE UNIT #
021612          CLC
021614          ROL      WORK
021620          BCC      1$
021622          HLT      !AS        ;WHY DID WE INT WITH NO ATA???
021624  2$:   MOV      R2,@RSCS2   ;GET DRIVE
021630          CMP      #110600,@RSDS ;DID PIP CLEAR?
021636          BEQ      +4         ;YES
021640          HLT      !DS!AS    ;PIP BIT DID NOT CLEAR
021642          CMP      #104230,@RSCS1 ;DID SC SET?
021650          BEQ      +4         ;YES
021652          HLT      !AS!DS    ;SC DID NOT SET
021654          DEC      SAVEE     ;COUNT # OF INT
021660          BEQ      WTDV1     ;DONE YET?
021662          MOV      WORK,@RSAS ;CLEAR AS
021670          MOV      #100,@RSCS1 ;SET IE
021676          MOV      #WTDV2,(SP) ;RETURN TO WAIT
021702          RTI
021704  WTDV1: MOV      #340,@#PS
021712          MOV      #500,SP    ;CLEAR STACK
021716          MOV      @RSVCP5,@RSVEC ;RESTORE INT VECTOR
021724          CLR      @RSVCP5
021730  MODDON: SCOPE
021732          MOV      TIMSV,TIMES ;RESTORE LOOP COUNT
021740          MOV      #340,@#PS  ;RESTORE PS
021746          MOV      #1,ICNT    ;FUZE TEST NUMBERS
021754  OUT:   JMP      @#TRYNX    ;TEST NEXT DRIVE
          .SBTTL                ;TEST NEXT DRIVE
                                ;DONE - BELL AND SCOPE ROUTINE

021760  DONE:  SCOPE
021762          ADD      #1,PCNT+2  ;TERMINATIONG SCOPE FOR LOOPING
021770          ADC      PCNT        ;ADD 1 TO THE PASS COUNT
021774          BIT      #SW10,@SWR  ;MAKE IT DOUBLE PREC.
022002          BNE      4$
022004          TYPE      +2        ;RING THE BELL?
022014  4$:   MOV      @#42,RO     ;NO!
022020          BEQ      $END1      ;ASCIZ <BELL><177>
022022          RESET
022024  $ENDAD: JSR      7,(0)      ;GET MONITOR ADDRESS
022026          JMP      240,240    ;SAVE ROOM FOR ACT11
022034  $END1: JMP      @#BEGIN    ;IF NONE
                                ;GO TO MONITOR
                                ;RETURN

022040  .TBIT:  0                  ;T BIT FLAG

022042  DVTAB: .BLKW  10

```

.SBTTL STYPE - TTY TYPEOUT ROUTINE

: THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE  
 : CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ADR" - TYPES THE  
 : MESSAGE STARTING IN LOCATION "ADR:" 2) "TYPE CHAR" - TYPES  
 : THE ASCII "CHAR" AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES  
 : THE MESSAGE WHICH IS IN LINE ASCII. THE FILLER CHARACTER WHICH IS  
 : TYPED AFTER A LINE FEED IS IN FILCHR AND THE NUMBER OF FILLERS  
 : IS IN FILCHR+1.

022062	.TYPE:	MOV	R4,-(6)	:SAVE R4
022064		MOV	R5,-(6)	:SAVE R5
022066		MOV	24(6),R5	:GET ADDRESS TO BE TYPED
022072		BIT	#177400,R5	:IS IT A TYPED?
022076		BNE	1\$	:NO
022100		MOV	4(6),R5	:GET ADDRESS OF CHARACTER
022104	1\$:	TSTB	(R5)	:TERMINATOR?
022106		BEQ	2\$	:GET OUT IF SO
022110		CMPB	#12,(R5)	:IS THE CHAR A LINE FEED
022114		BNE	4\$	:NO - GET OUT
022116		MOVB	FILCHR+1,R4	:GET THE FILL COUNT
022122	5\$:	MOVB	FILCHR,@TPB	:TYPE A FILLER
022130		TSTB	@TPS	:DONE YET?
022134		BPL	.-4	:NO - WAIT
022136		DEC	R4	:DEC COUNT
022140		BNE	5\$	:LOOP UNTIL 0
022142	4\$:	MOVB	(R5)+,@TPB	:LOAD AND TYPE THE CHARACTER
022146		TSTB	@TPS	:IS THE PRINTER READY
022152		BPL	.-4	:WAIT UNTIL IT IS
022154		BR	1\$	:GET THE NEXT CHARACTER
022156	2\$:	MOV	24(6),-(6)	:GET ADDRESS TO BE TYPED
022162		ADD	#2,6(6)	:ADD 2 TO THE ADDRESS
022170		CMP	(6)+,4(6)	:IS IT .+2?
022174		BNE	3\$	:NO
022176		ADD	#2,R5	:ADD 2 TO THE ADDRESS
022202		BIC	#1,R5	:BACK UP TO AN EVEN BYTE
022206		MOV	R5,4(6)	:RESTORE ADDRESS
022212	3\$:	MOV	(6)+,R5	:RESTORE R5
022214		MOV	(6)+,R4	:RESTORE R4
022216		RTI		:RETURN

.SBTTL \$SCOPE - SCOPE LOOP HANDLER

: THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR  
: LOOPING, AND THE DISPLAYING OF THE TEST NUMBER  
: \$SCOPE IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND  
: RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"

022360	.SCOPE:	KBOIN	BIT	\$SW8,2SWR	: GO CHECK FOR 1G
022361			BEO	1	: LOOP ON SPEC TEST?
022362			CHPB	\$SWR,ICNT	: NO LOOP ON SPEC. TEST
022363			BEO	.OVER	: ON RIGHT TEST? *SW7-0*
18:			BIT	\$SW14,2SWR	: NOT RIGHT TEST
022365			BNE	.KIT	: LOOP ON TEST?
022366			BR	\$SW8	: LOOP ON TEST IS SET
022367			MOV	\$SW4,-(6)	: SKIP - NOP FOR XOR TESTER
022368			MOV	\$SW8,284	: PUSH 284 ON STACK
022369			TST	\$SW17,7060	: SET FOR TIMEOUT
022370			MOV	(6)+,284	: ERROR ON XOR?
022371			BR	.SVLAD	: POP STACK INTO 284
48:			CHPB	(6)+,(6)+	: NO ERROR - GO TO NEXT TEST
022373			MOV	(6)+,284	: CLEAR STACK
022374			BR	.KIT	: POP STACK INTO 284
38:			BIT	\$SW11,2SWR	: ERROR - LOOP ON TEST
022376			BNE	.SVLAD	: KILL ITERATIONS
022377			TSTB	ICNT+1	: YES - KILL ITERATIONS
022378			BEO	\$SW8	: FIRST ONE?
022379			CHPB	TIMES,ICNT+1	: BRANCH IF FIRST
022380			BGT	.KIT	: DONE?
28:			MOV	\$SW1,ICNT+1	: BRANCH IF NOT
022382	.SVLAD:		INCB	ICNT	: FIRST ITERATION
022383			MOV	(6),LAD	: COUNT TEST NUMBERS
022384			MOV	ICNT,2DISPLAY	: SAVE LOOP ADDRESS
022385			RTI		: DISPLAY TEST NO. AND ITERATION COUNT
					: RETURN
022364	.KIT:		INCB	ICNT+1	: INC THE ITERATION COUNT
022370	.OVER:		MOV	ICNT,2DISPLAY	: SET UP DISPLAY
022376			TST	LAD	: FIRST ONE?
022402			BEO	.SVLAD	: YES
022404			MOV	LAD,(6)	: FUDGE RETURN ADDRESS
022410			RTI		: FIXES PS
022412	TIMES:	100			: RUN 100 TIMES



.SBTTL SOCIAL - OCTAL TYPEOUT ROUTINE  
:THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE  
:ALL 6 CHARACTERS, SUPPRESS LEADING ZERO'S, OR TYPE THE  
:16 BITS. IT IS CALLED VIA THE TYOCT, TYBIT, OR TYOCS MACRO'S.

```

022754 .TYPEB: MOV #170101..PR ;SET BIT FLAG AND 16 CHARACTER COUNT
022756 BR .PTIT ;NOW TYPE IT IN BIT FORM
022758 .TYPEO: MOVB #1..PR ;SET ZERO FILL SWITCH
022760 BR ;SKIP
022762 .TYPES: CLR .PR ;SUPPRESS LEADING ZERO'S
022764 MOVB #-6..PR+1 ;SET COUNT
022766 .PTIT:
022768 MOV R4, -(6) ;PUSH R4 ON STACK
022770 MOV R5, -(6) ;PUSH R5 ON STACK
022772 MOV 10(6), R5 ;GET THE DATA
022774 MOV #.PR+2, R4 ;SET POINTER TO FIRST ASCII CHAR.
022776 CLRB (4) ;CLEAR FIRST BYTE
022778 BR PRF ;ROTATE FIRST BIT
022780 .PRL: CLRB (4) ;CLEAR BYTE OF CHARACTER
022782 BIT #100..PR ;BIT TYPING MODE?
022784 BNE .PRF ;YES - SKIP 2 ROTATES
022786 ROL R5 ;ROTATE BIT INTO C
022788 ROLB (4) ;PACK IT
022790 ROL R5 ;ROTATE BIT INTO C
022792 ROLB (4) ;PACK IT
022794 .PRF: ROL R5 ;ROTATE BIT INTO C
022796 ROLB (4) ;PACK IT
022798 TSTB (4) ;IS IT ZERO?
022800 BRG .+6 ;SKIP INC
022802 INCB .PR ;SET FILL SWITCH
022804 TSTB .PR ;CHECK FILL SWITCH
022806 BRG .+6 ;SKIP BITSET
022808 BICB #'0'(4), .PR+1 ;MAKE INTO ASCII CHAR
022810 INCB .PR ;INC COUNT
022812 BNE .PRL ;REPEAT
022814 CNB #.PR+2, R4 ;EMPTY BUFFER?
022816 BNE .+6 ;SKIP IF NOT
022818 MOVB #'0'(4), .PR ;LOAD 1 ZERO
022820 CLRB (4) ;NULL TERMINATOR
022822 TYPE .PR+2 ;TYPE IT
022824 MOV (6)+, R5 ;POP STACK INTO R5
022826 MOV (6)+, R4 ;POP STACK INTO R4
022828 MOV 2(6), 4(6) ;GET RID OF
022830 MOV (6)+, (6) ;DATA WORD
022832 RTI ;RETURN
022742 .PR: .BLKW 12 ;COUNT, SWITCH, AND OUTPUT BUFFER

```

.SBTTL \$POWER - POWER DOWN AND UP ROUTINES

: THIS IS THE POWER FAIL ROUTINE WHICH WILL SAVE ALL  
 : THE GENERAL REGISTERS AND USER DEFINED REGISTERS THEN  
 : WAIT FOR POWER TO GO DOWN AND BE RESTORED.  
 : IF THERE ISN'T ENOUGH TIME FOR SAVING ALL THE REGISTERS,  
 : THE PROGRAM WILL HALT AT '.ILLUP'.

```

022766 .POWER: MOV      #.ILLUP, @.PUVEC : SET FOR FAST UP
022774      MOV      #340, @.PUVECS+2 : PRIO:7
023002      MOV      R0, -(6) : PUSH R0 ON STACK
023004      MOV      R1, -(6) : PUSH R1 ON STACK
023006      MOV      R2, -(6) : PUSH R2 ON STACK
023010      MOV      R3, -(6) : PUSH R3 ON STACK
023012      MOV      R4, -(6) : PUSH R4 ON STACK
023014      MOV      R5, -(6) : PUSH R5 ON STACK
023016      MOV      SP, .SAVR6 : SAVE SP
023022      MOV      #.POWUP, @.PUVEC : SET UP VECTOR
023030      HALT : WAIT FOR PF

023032 .POWUP: MOV      .SAVR6, SP : GET SP
023036      CLR      R1 : WAIT LOOP FOR THE TTY
023040 15: INC      R1 : WAIT FOR THE INC
023042      BNE     15 : OF WORD
023044      MOV      (6)+, R5 : POP STACK INTO R5
023046      MOV      (6)+, R4 : POP STACK INTO R4
023050      MOV      (6)+, R3 : POP STACK INTO R3
023052      MOV      (6)+, R2 : POP STACK INTO R2
023054      MOV      (6)+, R1 : POP STACK INTO R1
023056      MOV      (6)+, R0 : POP STACK INTO R0
023060      MOV      #.POWER, @#24 : SET UP THE POWER DOWN VECTOR
023066      MOV      #340, @#26 : PRIO:7
023074      TYPE : .ASCIZ (<15>(<12>)"POWER"
023110      RTI : RETURN

023112 .ILLUP: HALT : THE POWER UP SEQUENCE WAS STARTED
023114      BR      -2 : BEFORE THE POWER DOWN WAS COMPLETE

023116 .SAVR6: 0 : PUT THE SP HERE
023120 .PUVEC: 24, 26 : POWER UP VECTOR

```

# F08

.SBTTL                    \$RDOCT - OCTAL INPUT ROUTINE

: THIS ROUTINE CALLS RDLIN, INPUTS A LINE FROM THE TTY AND CONVERTS  
 : IT INTO AN OCTAL NUMBER WHICH IS THE FIRST WORD ON THE STACK.

```

023231 .RDOCT: MOV      (6) -(6)      : MOVE THE PC
023232      MOV      4(6), 2(6)      : MOVE THE PS
023233      MOV      R1, -(6)        : PUSH R1 ON STACK
023234      MOV      R2, -(6)        : PUSH R2 ON STACK
023235      MOV      R3, -(6)        : PUSH R3 ON STACK
023236 45:   RDLIN                      : READ A LINE INTO INPUT
023237      CLR      R1              : INIT DATA WORD
023238      CLR      CTN             : CLEAR COUNT WORD
023239      MOV      #INPUT, R3      : INIT POINTER
023240 15:   MOV      (3)+, R2        : GET A BYTE
023241      CMP      #15, R2        : WAS IT A CR?
023242      BEQ      25              : GET OUT IF YES
023243      CMP      #0, R2          : CHECK FOR 0 OR GREATER
023244      BGT      35              : ERROR - LESS THAN 0
023245      CMP      #7, R2          : CHECK FOR 7 OR LESS
023246      BGT      35              : ERROR - GREATER THAN 7
023247      ROR      R2              : GET
023248      ROR      R2              : INTO
023249      ROR      R2              : POSITION
023250      ROL      R1              : FIRST BIT
023251      ROL      R2              : GET
023252      ROL      R1              : SECOND BIT
023253      ROL      R2              : GET
023254      ROL      R1              : THIRD BIT
023255      INC      CTN             : YES HE TYPED SOMETHING
023256      BR      15              : LOOP
023257 25:   MOV      R1, 12(6)      : SAVE THE RESULT
023258      MOV      (6)+, R3        : POP STACK INTO R3
023259      MOV      (6)+, R2        : POP STACK INTO R2
023260      MOV      (6)+, R1        : POP STACK INTO R1
023261      R*1                      : RETURN
023262
023263 35:   TYPE      45+2           : .ASCIZ "???(15)(12)"
023264      BR      45+2             : TRY AGAIN
  
```

.SBTTL SRDLIN - TTY INPUT ROUTINE

: THIS ROUTINE INPUTS A LINE TERMINATED BY A RETURN INTO ADDRESS  
 : INPUT AND RETURNS A LINE FEED. THE BUFFER HAS A NULL TERMINATOR  
 : INSTEAD OF THE RETURN. RUBOUTS ARE HANDLED BY RETYPING  
 : THE LINE. BUFFER OVERFLOW ERRORS LIKE A RUBOUT.

```

023256 .RDLIN: MOV R5, -(6) ;SAVE R5
023260 1$: MOV @INPUT, R5 ;GET ADDRESS
023264 2$: CMP @INPUT+16., R5 ;BUFFER FULL?
023270 BEQ 4$ ;YES - TYPE "?"
023272 YSTB @177560 ;WAIT FOR
023276 BPL -4 ;A CHARACTER
023300 MOVB @177562.(5) ;GET CHARACTER
023304 BICB @200.(5) ;GET RID OF JUNK
023310 CMPB @25.(5) ;IS IT A ?U
023314 BNE 5$ ;BRANCH IF NOT
023316 TYPE .+2 ;.ASCIZ "?U"(15)(12)
023330 BR 1$ ;START OVER
023332 5$: CMPB @177.(5) ;IS IT A RUBOUT
023336 BNE 3$ ;SKIP IF NOT
023340 4$: TYPE .+2 ;.ASCIZ "?"(15)(12)
023340 BR 1$ ;ZAP THE BUFFER AND LOOP
023350 3$: MOVB (5), #0 ;SET UP FOR TYPING
023352 TYPE 3$+2 ;ECHO IT
023356 CMPB @15.(5)+ ;CHECK FOR RETURN
023362 BNE 2$ ;LOOP IF NOT RETURN
023366 TYPE 12 ;TYPE A LINE FEED
023374 MOV (6)+, R5 ;RESTORE R5
023376 RTI ;RETURN
    
```

023400 INPUT: .BLKB 16. ;TTY INPUT AREA  
 .SBTTL STRAP - TRAP HANDLER

: THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE  
 : SUBROUTINE. THE CALL IS A "TRAP+N" WHERE N IS A MULTIPLE OF 2.  
 : THE "SET" MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO  
 : FOLLOW THIS MACRO.

```

023420 .TRAP: MOV (6), -(6) ;GET ADDRESS OF TRAP +2
023422 SUB #2, (6) ;MAKE IT ADDRESS OF TRAP
023426 MOV @ (6), (6) ;GET TRAP INSTRUCTION
023432 ADD #.TRAP+2-TRAP, (6) ;GET DATA AND MAKE IT AN OFFSET
023436 .TRAP: MOV @ (6)+, PC ;GO TO PROPER SUBROUTINE
    
```

023440	.SCOPE	:SCOPE	=	TRAP+0	(104400)
023442	.TYPE	:TYPE	=	TRAP+2	(104402)
023444	.TYPE0	:TYPE0	=	TRAP+4	(104404)
023446	.TYPES	:TYPES	=	TRAP+6	(104406)
023450	.RDOCT	:RDOCT	=	TRAP+10	(104410)
023452	.RDLIN	:RDLIN	=	TRAP+12	(104412)
023454	.CLARK	:CLARK	=	TRAP+14	(104414)
023456	.KBDIN	:KBDIN	=	TRAP+16	(104416)
023460	.SUSWR	:SUSWR	=	TRAP+20	(104420)
023462	.CNTLU	:CNTLU	=	TRAP+22	(104422)

:ROUTINE TO ALLOW THE OPERATOR TO SET BITS  
:IN THE I/O REGISTERS VIA THE SWITCH REGISTER

:WORD COUNT REGISTER  
023464 SRSWC: MOV #500,SP :SET UP STACK FOR TRAP CALL  
023470 IS: KBDIN :CHECK THE WORLD  
023472 :MOV SWR, SRSWC :MOV SWR INTO WORD COUNT REG  
023500 :MOV SRSWC, @DISPLAY :DISPLAY IN LIGHTS  
023506 BR IS

:CURRENT ADDRESS REGISTER  
023510 SRSBA: MOV #500,SP :INIT THE STACK  
023514 IS: KBDIN :CTW  
023516 :MOV SWR, SRSBA :MOV SWR INTO CURRENT ADDR REG  
023524 :MOV SRSBA, @DISPLAY :SHOW IN LIGHTS  
023532 BR IS

:DISK ADDRESS REGISTER  
023534 SRSDA: MOV #500,SP :INIT THE STACK  
023540 IS: KBDIN :CTW  
023542 :MOV SWR, SRSDA :MOV SWR INTO DISK ADDR REG  
023550 :MOV SRSDA, @DISPLAY :SHOW IN LIGHTS  
023556 BR IS

:DRIVE STATUS REGISTER  
023560 SRSDS: MOV #500,SP :INIT THE STACK  
023564 IS: KBDIN :CTW  
023566 :MOV SWR, SRSDS :MOV SWR INTO DRIVE STATUS  
023574 :MOV SRSDS, @DISPLAY :SHOW IN LIGHTS  
023602 BR IS

:DRIVE ERROR REGISTER  
023604 SRSER: MOV #500,SP :INIT THE STACK  
023610 IS: KBDIN :CTW  
023612 :MOV SWR, SRSER :LOAD ER REG  
023620 :MOV SRSER, @DISPLAY :DISPLAY IT IN LIGHTS  
023626 BR IS :LOOP

:WATCH LOOK AHEAD REGISTER  
023630 SRSLA: MOV SRSLA, @DISPLAY :SHOW IN LIGHTS  
023636 BR SRSLA

```

:RSCS2 REGISTER
023640 SRCS2: MOV #500,SP ;INIT THE STACK
023644 IS: KBDIN ;CTW
023646 ;JV ;SWR,SRCS2 ;LOAD CS2
023654 MOV ;SRCS2,DISP ;DISPLAY IT
023662 BR IS

:RSAS REGISTER
023664 SRAS: MOV #500,SP ;INIT THE STACK
023670 IS: KBDIN ;CTW
023672 ;MV ;SWR,RSAS ;LOAD RSAS
023700 MOV ;RSAS,DISP ;DISPLAY IT
023706 BR IS

:RSMR REGISTER
023710 RSMR: MOV #500,SP ;INIT THE STACK
023714 IS: KBDIN ;CTW
023716 ;MV ;SWR,RSMR ;LOAD RSMR
023724 MOV ;RSMR,DISP ;DISPLAY IT
023732 BR IS

:DISK CONTROL STATUS REGISTER
023734 SRCS1: MOV #500,SP ;INIT THE STACK
023740 IS: KBDIN ;CTW
023742 ;MV ;340,PS ;LOCK UP INTERRUPTS
023750 ;MV ;177777,RSWC ;SET WORD COUNT -1 WORD
023756 ;MV ;OBUSV,RSBA ;SET UP CURRENT ADDRESS
023764 ;MV ;SWR,SRCS1 ;MOV SWR INTO CONTROL REG
023772 BIT ;BIT0,SRCS1 ;IS FUNCTION BITS SET
024000 BEQ IS ;FUNCTION BITS NOT SET
024002 ;TSTB ;SRCS1 ;TEST FOR DISK READY
024006 ;BPL ;25 ;DISK STILL NOT READY
024010 ;BR ;IS ;DISK NOT BUSY SECT NEW RA
    
```

# JOB

```

: THIS ROUTINE GIVES THE OPERATOR THE ABILITY TO
: SELECT DA, WC, UNIT # AND DESIRED PATTERN. PATTERN = NUMBER TYPED
: WITH SW12 SET THE PROGRAM WILL LOOP ON A READ WITH LOC OUTBUF+2 AS
: THE BA ADDR. WITH BIT12 0 IN THE SWR THE PROGRAM
: WILL WRITE WITH OUTBUF AS THE BA ADDR. BAI IS ALWAYS SET
: SWITCHES 0 TO 11 WILL DETERMINE THE DA
: EXAMPLE:
  
```

```

TYPE UNIT # 5
TYPE POSITIVE (OCTAL) WC 64
TYPE PATTERN DESIRED 1252525
  
```

```

024012  TASEL:  MOV      #500,SP          ;SET STACK
024016      NOP
024020      TYPE      ..+2          ;.ASCIZ <15><12>"TYPE UNIT # "
024044      RDOCT
024046      MOV      (6)+,UNNUM      ;.ASCIZ <15><12>"TYPE POSITIVE (OCTAL) WC "
024052      TYPE      ..+2
024112      RDOCT
024114      MOV      (6)+,WORK
024120      COM      WORK
024124      TYPE      ..+2          ;.ASCIZ <15><12>"TYPE PATTERN DESIRED "
024160      RDOCT
024162      MOV      (6)+,OUTBUF
024166      BIC      #BIT0,SWI      ;CLEAR THE BEENHEREBIT
024174      SUSWR
024176      TK1:   MOV      @SWR,WORK2 ;INIT SWITCHLESS
024204      TK2:   CLDRK          ;SAVE SWR
024206      BIS      #BIT3,@RSCS2   ;CLEAR ALL RS REG
024214      MOV      @SWR,WORK1     ;SET BAI
024222      BIC      #17000,WORK1   ;GET SWR FOR DSK ADDR
024230      TKKS:  MOV      WORK1,@RSDA ;CLEAR UNIT #
024236      MOV      WORK,@RSCW     ;LOAD THE DA
024244      BIT      #BIT12,@SWR    ;LOAD WORD COUNT
024252      BEQ      WTE           ;READ?
024254      MOV      #OUTBUF+2,@R5BA ;NO
024262      MOV      #71,@RSCS1     ;LOAD CURRENT ADDRESS
024270      WT:   TSTB      @RSCS1   ;GO AND READ
024274      BPL      -4            ;TEST FOR READY
024276      BR      SWRCHG
024300      WTE:   MOV      #OUTBUF,@R5BA
024306      MOV      #61,@RSCS1
024314      BR      WT
024316      SWRCHG: MOV      @RSCS1,@DISPLAY ;DISPLAY CS1
024324      TST      @RSCS1         ;ANY ERRORS?
024330      BPL      IS            ;NO
024332      HLT      !DA!WC
024334      IS:   KBDIN
024336      CMP      @SWR,WORK2     ;CHECK FOR NEW VALUE
024344      BNE      TK1           ;DID SWR CHANGE?
024346      BR      TK2           ;YES
                                ;NO
  
```

\*\*\*\*\*  
:TEST 122 WRITE LOCK TEST  
\*\*\*\*\*  
TST122: SCOPE

024350

024352

024352

024416

024424

024426

024430

024432

024434

024442

024444

024450

024456

024464

024472

024500

024504

024506

024560

024562

024570

024576

024600

024602

024610

024616

024620

024622

024662

024664

024672

024674

024676

024704

024712

024714

024716

024756

024760

024766

024770

024772

025000

025006

025010

025012

025052

WRTLCK:

TYPE

CMP

BEQ

HALT

BR

CNTLU

MOV

CLRDK

CLR

MOV

MOV

MOV

TSTB

BPL

TYPE

HALT

MOV

CMP

BEQ

HALT

MOV

MOV

CMP

BEQ

HALT

TYPE

HALT

CMP

BEQ

HALT

MOV

CMP

BEQ

HALT

TYPE

HALT

..+2

#SWREG,SWR

7\$

8\$

#SWR,UNNUM

OUTBUF

#OUTBUF,#RSBA

#7700,#RSDA

#-1,#RSC

#61,#RSCS1

#RSCS1

6\$

..+2

#0,#RSDA

#14600,#RSDS

..+4

!DS!DA

#100,#RSDA

#10600,#RSDS

..+4

!DS!DA

..+2

#14600,#RSDS

..+4

!DS!DA

#300,#RSDA

#10600,#RSDS

..+4

!DS!DA

..+2

#14600,#RSDS

..+4

!DS!DA

..+2

;.ASCIZ <15><12>"LOAD SW WITH UNIT # AND CONT"

;CHECK IF SWITCHLESS CPU

;GO AROUND TRAP CALL

;GET SWREG VALUE

;GET UNIT #

;CLEAR ALL REG

;PUT A 0 INTO DATA BUFFER

;SETUP REG TO

;TO A WRITE

;WAIT FOR DONE

;.ASCIZ <15><12>"SET WRITE LOCK ENABLE AND CONTINUE"

;DS SHOULD=14600

;DS SHOULD=10600

;.ASCIZ <15><12>"SET WRT LOC SW 0 AND CONT"

;DS SHOULD=14600

;DS SHOULD=10600

;.ASCIZ <15><12>"SET WRT LOC SW 1 AND CONT"

;DS SHOULD=14600

;DS SHOULD=10600

;.ASCIZ <15><12>"SET WRT LCK SW 2 AND CONT"

```

025054 4S:  CMP      #14600, @RSDS
025062      BEQ      +4
025064      HLT      !DS!DA      ; DS SHOULD=14600
025066      MOV      #1700, @RSDA
025074      CMP      #10600, @RSDS
025102      BEQ      +4
025104      HLT      !DS!DA      ; DS SHOULD=10600
025106      TYPE    ..+2      ;.ASCIZ <15><12>"SET WRT LCK SW 3 AND CONT"
025146 HALT:  HALT
025150      CMP      #14600, @RSDS
025156      BEQ      +4
025160      HLT      !DS!DA      ; DS SHOULD=14600
025162      MOV      #3700, @RSDA
025170      CMP      #10600, @RSDS
025176      BEQ      +4
025200      HLT      !DS!DA      ; DS SHOULD=10600
025202      TYPE    ..+2      ;.ASCIZ <15><12>"SET WRT LCK SW 4 AND CONT"
025242      HALT
025244      CMP      #14600, @RSDS
025252      BEQ      +4
025254      HLT      !DS!DA      ; DS SHOULD=14600
025256      MOV      #6000, @RSDA
025264      CMP      #10600, @RSDS
025272      BEQ      +4
025274      HLT      !DS!DA      ; DS SHOULD=10600
025276      TYPE    ..+2      ;.ASCIZ <15><12>"SET WRT LCK SW 5 AND CONT"
025336      HALT
025340      CMP      #14600, @RSDS
025346      BEQ      +4
025350      HLT      !DS!DA      ; DS SHOULD=14600
025352      MOV      #7700, @RSDA
025360      CMP      #14600, @RSDS
025366      BEQ      +4
025370      HLT      !DS!DA      ; DS SHOULD=14600
025372      MOV      #-1, @OUTBUF      ; PUT A 1 INTO DATA BUFFER
025400      MOV      @OUTBUF, @RSBA      ; SETUP REG TO
025406      MOV      #7700, @RSDA      ; TO A WRITE
025414      MOV      #-1, @R5WC
025422      MOV      #61, @R5CS1      ; TRY TO WRITE
025430 7S:  TSTB    @R5CS1      ; WAIT FOR DONE
025434      BPL      7S
025436 5S:  TSTB    @R5CS1      ; WAIT FOR READY
025442      BPL      5S
025444      CMP      #154600, @RSDS
025452      BEQ      +4
025454      HLT      !DS!DA      ; DS SHOULD=154600
    
```

```

025456      CMP      #4000, @RSER
025460      BEQ      +4
025466      HLT      !DS!DA          ; ER SHOULD=4000
025470      CLRDK   ; CLEAR ALL REG
025472      CMP      #14600, @RSDS  ; DID WLE CLEAR?
025500      BEQ      +4              ; NO
025502      HLT      !DS              ; A CLEAR SHOULD NOT CLEAR WLE
025504      CLR      OUTBUF          ; PUT A 0 INTO DATA BUFFER
025510      MOV      @OUTBUF, @RSBA  ; SETUP REG TO
025516      MOV      #7700, @RSDA   ; TO A WRITE
025524      MOV      #-1, @ASWC
025532      MOV      #71, @RSCS1
025540      BS:    TSTB   @RSCS1     ; DO A READ TO SEE IF DISK DID
025544      BPL      BS              ; ACTUALLY GET WRITTEN ON TO
025546      TST      OUTBUF          ; WAIT FOR DONE
025552      BEQ      +4              ; IS DATA STILL 0 ON THE DSK?
025554      HLT      !DS              ; YES
025556      TYPE    ,.+2            ; NO DSK DID GET WRITTEN ONTO WITH WLE SET
025614      HALT
025616      CMP      #10600, @RSDS  ; DID WLE CLEAR
025624      BEQ      +4              ; YES
025626      HLT      !DS              ; NO

:ROUTINE FOR FINDING MEMORY ON "B" PORT
025630      FINDM:  MOV      #MAXREF, 4 ; SET UP I/O BUS TRAP
025636      MOV      #340, 6          ; SET PS
025644      BIT      #BIT12, @SWR     ; INHIBIT OBUFSV FROM CHANGING?
025652      BNE      EXREF           ; YES
025654      ADD      #20000, OBUFSV   ; ADD 4 K
025662      EXREF:  TST      @OBUFSV  ; LEGAL LOC ? IF NO TRAPS
025666      MOV      #6, 4           ; RESTORE I/O BUS TRAP
025674      CLR      6
025700      CMP      #177446, OBUFSV ; TEST FOR GREATER THEN 28K
025706      BLO      MAXRF1         ; YES
025710      JMP      WRTSTB         ; RETRY WRITING
025714      MAXREF:  CMP      (6)+, (6)+ ; CLEAR STACK
025716      MAXRF1:  MOV      #6, 4  ; RESTORE I/O BUS TRAP
025724      CLR      6
025730      MOV      @OUTBUF, OBUFSV ; RESTORE ORIGINAL VALUE
025736      BIT      #BIT13, @SWR     ; INHIBIT TYPEOUT?
025744      BNE      1$             ; YES
025746      BIT      #BIT13, ONCEE   ; DID WE TYPE THIS YET?
025754      BNE      1$             ; YES
025756      TYPE    ,.+2            ; .ASCIZ <15><12>"COULD NOT FIND MEMORY ON -B- PORT"<15><12>

: EVEN
026030      1$:    BIS      #BIT13, ONCEE ; SET TYPE FLAG
026036      SCOPE   ; UPDATE
026040      SCOPE   ; TEST NUMBER?
026042      JMP      NXM            ; CONT TESTING

: CLEAR ALL DISK REGISTERS
026046      CLRDK:  MOV      #40, @RSCS2 ; CLEAR ALL DSK REG
026054      MOV      UNNUM, @RSCS2    ; GET UNIT NUMBER
026062      RTI

```

:ERROR TYPTXTOUT ROUTINE

```

026064 RSREG: TST      .HLTCT      ; SHOULD WE TYPTXT GOOD AND BAD
026070      BNE      8$          ; NO
026072      TYPE     .+2         ; .ASCIZ " BAD="
026104      MOV      BAD,-(6)    ; PUT BAD ON STACK
026106      TYPE0    ; TYPE STACK IN OCTAL
026110      TYPE     .+2         ; .ASCIZ " GOOD="
026124      MOV      GOOD,-(6)  ; PUT GOOD ON STACK
026126      TYPE0    ; TYPE STACK IN OCTAL
026130      JMP      PTDONE     ; GET OUT
026134 8$:   TYPE     .+2         ; .ASCIZ " CS1="
026146      MOV      ARSCS1,-(6) ; PUT ARSCS1 ON STACK
026152      TYPE0    ; TYPE STACK IN OCTAL
026154 1$:   TYPE     .+2         ; .ASCIZ " ER="
026166      MOV      ARSER,-(6)  ; PUT ARSER ON STACK
026172      TYPE0    ; TYPE STACK IN OCTAL
026174 2$:   TYPE     .+2         ; .ASCIZ " CS2="
026206      MOV      ARSCS2,-(6) ; PUT ARSCS2 ON STACK
026212      TYPE0    ; TYPE STACK IN OCTAL
026214      BIT      #200,.HLTCT ; TYPTXT SECOND SET ?
026222      BNE      SEEC       ; YES
026224      BIT      #AS,.HLTCT ; TYPTXT ER ?
026232      BEQ      3$         ; NO
026234      TYPE     .+2         ; .ASCIZ " AS="
026246      MOV      ARSAS,-(6)  ; PUT ARSAS ON STACK
026252      TYPE0    ; TYPE STACK IN OCTAL
026254 3$:   BIT      #BA,.HLTCT ; TYPTXT BUS ADDRESS
026262      BEQ      4$         ; NO
026264      TYPE     .+2         ; .ASCIZ " BA="
026276      MOV      ARSBA,-(6)  ; PUT ARSBA ON STACK
026302      TYPE0    ; TYPE STACK IN OCTAL
026304 4$:   BIT      #DA,.HLTCT ; TYPTXT DA ?
026312      BEQ      5$         ; NO
026314      TYPE     .+2         ; .ASCIZ " DA="
026326      MOV      ARSDA,-(6)  ; PUT ARSDA ON STACK
026332      TYPE0    ; TYPE STACK IN OCTAL
026334 5$:   BIT      #WC,.HLTCT ; TYPTXT WC?
026342      BEQ      6$         ; NO
026344      TYPE     .+2         ; .ASCIZ " WC="
026356      MOV      ARSWC,-(6)  ; PUT ARSWC ON STACK
026362      TYPE0    ; TYPE STACK IN OCTAL
    
```

```

026534 65: BIT B05..MLTCT :DRIVE STATUS
026536 BEQ PTDONE :NO
026538 TYPE A,+2 :ASCIZ "DS="
026540 MOV @RSDS,-(6) :PUT @RSDS ON STACK
026542 TYPE0 :TYPE STACK IN OCTAL
026544 JMP PTDONE :GET OUT
026546 SEEC: BIT B200..MLTCT :CLEAR COMMON BIT
026548 BIT B01..MLTCT :TYPTXT DRIVE TYPE?
026550 BEQ 95 :NO
026552 TYPE A,+2 :ASCIZ "DT="
026554 MOV @RSDT,-(6) :PUT @RSDT ON STACK
026556 TYPE0 :TYPE STACK IN OCTAL
026558 95: BIT B04..MLTCT :TYPTXT DATA BUFFER
026560 BEQ 105 :NO
026562 TYPE A,+2 :ASCIZ "DB="
026564 MOV @RSD8,-(6) :PUT @RSD8 ON STACK
026566 TYPE0 :TYPE STACK IN OCTAL
026568 105: BIT BMR..MLTCT :TYPTXT MN?
026570 BEQ 115 :NO
026572 TYPE A,+2 :ASCIZ "MR="
026574 MOV @RSMR,-(6) :PUT @RSMR ON STACK
026576 TYPE0 :TYPE STACK IN OCTAL
026578 115: BIT B1A..MLTCT :TYPTXT LA?
026580 BEQ PTDONE :NO
026582 TYPE A,+2 :ASCIZ "LA="
026584 MOV @RSLA,-(6) :PUT @RSLA ON STACK
026586 TYPE0 :TYPE STACK IN OCTAL
026588 PTDONE: BIS @BITIS,ONCEE :SET FORMD ERROR FLAG
026590 RTS PC

026576 WAITRY: CLR WORK :CLEAR COUNTER
026578 15: TSTB @RSCS1 :TEST READY
026580 BHI 25 :OK CONT
026582 INC WORK :UPDATE COUNTER
026584 TST WORK :DONE YET?
026586 BEQ 35 :READY DID NOT COME JP
026588 BR 15 :CONTINUE WAITING
026590 25: RDC 15, SP :UPDATE RETURN PC
026592 35: RTS PC :RETURN

```

:RANDOM DATA GENERATOR SUBROUTINE

```

026632 RANDOM: MOV LONUM, LOSAV
026634 MOV HINUM, HISAV
026636 RAND1: MOV LONUM, R0
026638 MOV HINUM, R4
026640 MOV R7, R3
026642 SHIFT: CLR R2
026644 ROL R0
026646 ROL R1
026648 DEC R3
026650 BNE SHIFT
026652 MOV LONUM, R0
026654 ROL R2
026656 MOV HINUM, R4
026658 ROL R2
026660 MOV R1, R0
026662 ROL R2
026664 MOV R2, R0
026666 ROL R2
026668 MOV R2, R0
026670 ROL R2
026672 MOV R2, R0
026674 ROL R2
026676 MOV R2, R0
026678 ROL R2
026680 MOV R2, R0
026682 ROL R2
026684 MOV R2, R0
026686 ROL R2
026688 MOV R2, R0
026690 ROL R2
026692 MOV R2, R0
026694 ROL R2
026696 MOV R2, R0
026698 ROL R2
026700 MOV R2, R0
026702 ROL R2
026704 MOV R2, R0
026706 ROL R2
026708 MOV R2, R0
026710 ROL R2
026712 MOV R2, R0
026714 ROL R2
026716 MOV R2, R0
026718 ROL R2
026720 MOV R2, R0
026722 ROL R2
026724 MOV R2, R0
026726 ROL R2
026728 MOV R2, R0
026730 ROL R2
026732 MOV R2, R0
026734 ROL R2
026736 MOV R2, R0
026738 ROL R2
026740 MOV R2, R0
026742 ROL R2
026744 MOV R2, R0
026746 ROL R2
026748 MOV R2, R0
026750 ROL R2
026752 MOV R2, R0
026754 ROL R2
026756 MOV R2, R0
026758 ROL R2
026760 MOV R2, R0
026762 ROL R2
026764 MOV R2, R0
026766 ROL R2
026768 MOV R2, R0
026770 ROL R2
026772 EXGEN: RTS
026774 LONUM: 0
026776 HINUM: 0
027000 LOSAV: 0
027002 HISAV: 0
027004 RANEND:

027004 .SUSWR: BIT #BIT0, SWI
027012 BNE XXX
027014 MOV 6, -(SP)
027020 MOV 4, -(SP)
027024 MOV #15, 4
027032 CMP #1, 2SWR
027040 BEQ 25
027042 BR 35
027044 15: CMP (SP)+, (SP)+
027046 25: MOV #SWREG, SWR
027054 MOV #DISPREG, DISPLAY
027062 35: CMP #SWREG, SWR
027070 BNE 45
027072 TEST 42
027076 BNE 45
027100 CNTRL

```

```

:SET UP R0 WITH 5 DIGITS LOW
:SET UP R1 WITH 5 DIGITS HIGH
:SET UP SHIFT COUNT
:CLR R2
:SHIFT R0 LEFT AND
:ROTATE CARRY INTO LSB OF R1 INTO
:ROTATE CARRY OUT OF R1 INTO R2
:DECREMENT R3
:CONTINUE SHIFT LOOP
:ROUN IN NUMBER TO MAKE X 129
:PROPOGATE CARRY
:ROUN IN NUMBER TO MAKE X 129
:PROPOGATE CARRY
:ROUN LOW CONSTANT
:PROPOGATE CARRIES
:PROPOGATE AGAIN
:ROUN HIGH CONSTANT
:PROPOGATE CARRY
:ROUN HIGHEST CONSTANT
:REPRIME R0 WITH HIGH DIGIT
:PROPOGATE CARRY
:PUT R0 BACK IN LONUM
:LOAD WC

:PUT R1 BACK IN HINUM
:HOLD HINUM FOR PROGRAM

:RETURN TO PROGRAM

:SAVE 6 ON STACK
:SAVE 4 ON STACK
:SET UP TRAP ADDRESS
:TEST 177570
:FAKE OUT
:HARDWARE AVAILABLE
:ADJUST STACK
:SET UP SOFTWARE REGISTERS

:1ST TIME THRU?
:NO CHANGE STILL 177570
:ANY XXDP OR ACT
:SWR=000000
:GET INITIAL SETTINGS

```

```

027102 4S:  MOV (SP)+.4      :REPLACE 4 FROM STACK
027106  MOV (SP)+.6      :REPLACE 6 FROM STACK
027112 XXX: BIS  #BIT0,SWI  :SET THE BEENHEREBIT
027120  RTI                    :ALL DONE

027122 SWI:  0
    
```

```

027124 .KBDIN: TST 42      :GOT XXDP OR ACT
027130  BNE OKT          :YES, GET OUT
027132  CMP #SWREG,SWR  :GOT SWITCH-LESS MACHINE?
027134  BNE OKT          :NO GET OUT
027140  TSTB #TKS       :HAVE A CHARACTER
027142  BPL OKT          :NO GET OUT
027150  MOV #TKB,.MSG   :
027156  BIC #177600,.MSG :STRIP OFF GARBAGE
027164  CTPB #7,.MSG    :DO WE HAVE A 'G'
027172  BNE OKT          :NO GET OUT
027174  TYPE ..+2      :.ASCIZ <15><12>"IG"
027206 .CNTLU:         :
027208  TYPE ..+2      :.ASCIZ <15><12>"SWR= "
027222  MOV SWREG,-(6) :PUT SWREG ON STACK
027224  TYPE          :TYPE STACK IN OCTAL
027230  TYPE ..+2      :.ASCIZ " NEW= "
027246  RDOCT         :
027250  MOV (SP)+,.MSG  :GET NEW VALUE OFF STACK
027254  TST CTN         :DID HE TYPE <CR> OF 000000?
027260  BEQ OKT        :DONT CHANGE IF <CR>
027262  MOV .MSG,SWREG  :CHANGE VALUE OF SWREG
027270  OKT: RTI        :ALL DONE-EXIT

027272 .MSG:  0
027274 CTN:  0
    
```

: TABLES FOR ILLEGAL FUNCTION TESTS

```

027276 ILLTAB: 3
027300      300
027302      302
027304      304
027306      306
027310      310
027312      312
027314      314
027316      316
027320      320
027322      322
027324      324
027326      326
027330      330
027332      332
027334      334
027336      336
027340 ILLTAB: 3
027342      342
027344      344
027346      346
027350      350
027352      352
027354      354
027356      356
027360      360
027362 OUTBUF: .BLKW 300
030162 INBUF:  .BLKW 300
030762 PARITY: 0      :USED FOR PARITY TEST 67
          .END

```

ADDCF 012120  
AS = 000100  
ATA = 100000  
ROB1 001166  
BA = 000020  
BAO = 000000  
BAI = 000010  
BAITST 007754  
BEGIN 001202  
BELL = 000007  
BITBA 004350  
BITCS2 004234  
BITST 004162  
BITWC 004300  
BITO 000001  
BIT1 000002  
BIT10 = 002000  
BIT11 = 004000  
BIT12 = 010000  
BIT13 = 020000  
BIT14 = 040000  
BIT15 = 100000  
BIT2 = 000004  
BIT3 = 000010  
BIT4 = 000020  
BIT5 = 000040  
BIT6 = 000100  
BIT7 = 000200  
BIT8 = 000400  
BIT9 = 001000  
BLOCK 010206  
BPORTT 001170  
CHKCDV 001650  
CLROK = 104414  
CNTLL = 104422  
CSTST 012050  
CS1 = 000001  
CS2 = 000200  
C\*N = 027274  
DA = 000004  
DAC = 001000  
DAOTST 015512  
DAOTT 015670  
DB = 000210  
DCK = 100000  
DCKTST 014362  
DISPLA 001030  
DISPRE 000174  
DKADR 015052  
DLT = 100000  
DONCS 012236  
DONE 021760  
DONEE 002016

DONINT 020756  
DRCLR 005754  
DRY = 000200  
DS = 000040  
DT = 000240  
DVNUM 001346  
DVTAB 022042  
ER = 000002  
ERRINT 020760  
ERRA = 040000  
ERRINT 021050  
ERRORS 001002  
ERRA 012002  
ERTST 016034  
EXGEN 026772  
EXREF 025662  
EXTRP 020410  
EXTTRP 020416  
EXTTST 017674  
EXTT1 020216  
EXT1 020420  
FILCHR 001014  
FINDM 025630  
FINTST 021064  
FLOTBA 003052  
FLOTDA 003352  
FLOTWC 003212  
FUNDO 016616  
GOOD = 000001  
HALT 025146  
HINUM 026776  
HISAV 027002  
HLT = 104000  
HLTADR 001012  
IAERR 015126  
ICNT 001000  
IE = 000100  
ILFDN 010762  
ILFDNE 011332  
ILFTB2 027340  
ILF67 011334  
ILLFUN 010764  
ILLTAB 027276  
ILLS1 010506  
INBUF 030162  
INCW 014574  
INPUT 023400  
INTDON 020640  
INTR4 020642  
INTR5 020536  
INT112 020632  
INT114 020744  
IR = 000100

KBOIN = 104416  
KIPARO = 172340  
KIPAR1 = 172342  
KIPAR2 = 172344  
KIPAR3 = 172356  
KIPARO = 172300  
KIPOR1 = 172302  
KIPOR2 = 172304  
KIPOR3 = 172316  
KIPOR7 = 000204  
LA = 001010  
LAD 001010  
LASTSC 017164  
LATDON 012046  
LATST 011724  
LBT = 002000  
LONUM 026774  
LOP1 003772  
LOP2 004042  
LOP3 004112  
LOSAY 027000  
MAXREF 025714  
MAXRF1 025716  
MEMOUT 020424  
MODDON 021730  
MODNUM 021066  
MPRO = 172100  
MR = 000220  
MULTII = 001316  
N = 000123  
NED = 010000  
NEDDON 015420  
NEDTST 015242  
NNOO 015502  
NNOOP 005244  
NNOOP21 005562  
NOOP 005120  
NOOP21 005424  
NOPAR 012442  
NOWGO 002022  
NXM 007526  
NXMTSM 010060  
OBUFSV 001102  
OKT 027270  
ONCEE 001160  
OOUT 010504  
OR = 000200  
OUT 021754  
OUTBUF 027362  
PARITY 030762  
PART 012240  
PARTST 011646  
PATST 016330  
PATYST 016770

PCNT 001004  
PGE = 002000  
PGTST 013414  
PGTRAP 020520  
PI3 = 020000  
PRTP 012422  
PRTP1 012424  
PS = 177776  
PSW = 177776  
PTDOME 026566  
QES 020440  
QESDON 020534  
QO = 000001  
RANDOM 026632  
RAN01 026646  
RANENO 027004  
RANNU 001150  
RANTS 003742  
ROLIN = 104412  
ROCT = 104410  
ROTST 006376  
ROTSTB 007154  
RMAT1 013614  
RMAT2 013772  
RMAT3 014102  
RMAT4 014212  
RSAS 001122  
RSBA 001112  
RSBAB 001146  
RSCS1 001104  
RSCS1B 001140  
RSCS2 001106  
RSCS2B 001142  
RSDA 001114  
RSDB 001126  
RSDS 001116  
RSDT 001132  
RSER 001120  
RSLA 001124  
RSMR 001130  
RSMRR 023710  
RSREG 026064  
RSVCPS 001136  
RSVEC 001134  
RSWC 001110  
RSWCB 001144  
RSWCWT 006626  
RSO4DT 001162  
RW = 000006  
SAVBAD 001100  
SAVEE 001172  
SC = 100000  
SCOPE = 104400

SECT 017330  
SEEC 026420  
SETPAT 016502  
SHIFT 026664  
SILO 004632  
SILOB 004422  
SILOFL 005004  
SRAS 023664  
SRCS2 023640  
SRCSBA 023510  
SRSCS1 023734  
SRSOA 023534  
SRSS 023560  
SRSER 023604  
SRSLA 023630  
SRSWC 023464  
SRO = 177572  
STAT 000230  
STTEST 001432  
SUSWR = 104420  
SWI 027122  
SWR 001026  
SWRCHG 024316  
SWREG 000176  
SW10 = 002000  
SW11 = 004000  
SW12 = 010000  
SW13 = 020000  
SW14 = 040000  
SW15 = 100000  
SW8 = 000400  
SW9 = 001000  
TIMES = 022412  
TIMSV 001164  
TKB 001022  
TKKS 024230  
TKS 001020  
TKSEL 024012  
TK1 024176  
TK2 024204  
TMEOUT 012234  
TPB 001024  
TPS 001016  
TRE = 040000  
TRY 001326  
TRYNY 001634  
TSTAGN 012266  
TSTNEM 010120  
TSTVEC 021336  
TST1 001314  
TST10 003050  
TST100 015050  
TST101 015124

TST102	015240	TST25	003616	TST6	002626	WC	" =	000010	.CNTLU	027206
TST103	015510	TST26	003644	TST60	010566	WCF	" =	004000	.HLT	022414
TST104	015666	TST27	003702	TST61	010702	WCFD	" =	013374	.HLTCT	022552
TST105	016032	TST3	003744	TST62	011322	WCFDDB	" =	012770	.ILLUP	023112
TST106	016326	TST30	003740	TST63	011164	WCFDDB	" =	013012	.KBOIN	027124
TST107	016500	TST31	004160	TST64	011164	WCFDDB	" =	013374	.KIT	022364
TST11	003102	TST32	004222	TST65	011172	WCFDDB	" =	013052	.MSG	027272
TST110	016614	TST33	004276	TST66	012204	WCFDDB	" =	013022	.OVER	022370
TST111	016766	TST34	004344	TST67	012204	WCFDDB	" =	012460	.POWER	022766
TST112	017162	TST35	004420	TST7	002786	WCFDDB	" =	012466	.POWER	022302
TST113	017326	TST36	004520	TST70	012456	WORK	" =	001174	.PR	022742
TST114	017672	TST37	004630	TST71	013200	WORK1	" =	001176	.PRF	022650
TST115	020436	TST4	002506	TST72	013242	WORK2	" =	001200	.PRL	022626
TST116	020534	TST40	005002	TST73	013612	WCKT	" =	006564	.PTIT	022606
TST117	020640	TST41	005116	TST74	013770	WCKT	" =	007350	.PUVEC	023120
TST12	003126	TST42	005242	TST75	014100	WTLCK	" =	024352	.QQ	" = 000024
*TST120	020756	TST43	005422	TST76	014210	WTTST	" =	006150	.ROLIN	023256
TST121	021064	TST44	005560	TST77	014360	WTTSTB	" =	006736	.ROOCT	023124
*TST122	024350	TST45	005752	TAGG	002276	WT	" =	024270	.SAVR6	023116
*TST13	003210	TST46	006146	TTHOUT	010502	WTDV	" =	021334	.SCOPE	022220
TST14	003242	TST47	006374	TTEC	021566	WTDV1	" =	021704	.SUSWR	027004
TST15	003266	TST5	002600	TYPE	104402	WTDV2	" =	021564	.SVLAD	022344
TST16	003350	TST50	006562	TYPEO	" = " = 104404	WTE	" =	024300	.SWOPT	" = 167400
TST17	003402	TST51	006734	TYPES	" = " = 104406	WLP	" =	000004	.TBIT	022040
TST2	002052	TST52	007152	UNCHK	001156	XXX	" =	027112	.TRAP	023420
TST20	003426	TST53	007346	UNITST	007544	ZERONE	" =	004524	.TRP	023436
*TST21	003470	TST54	007542	UNITSV	001154	SENDAD	" =	022024	.TYPE	022062
TST22	003512	TST55	007752	UNNUM	001152	SEND1	" =	022034	.TYPEB	022554
TST23	003534	TST56	010056	UP	" = 000000	.CLDRK	" =	030764	.TYPEC	022564
TST24	003560	TST57	010204	WAITRY	026576			026046	.TYPES	022574

. ABS. 030764 000

ERRORS DETECTED: 0

DZRSBF, DZRSBF /DOC=DZRSBF.SML, DZRSBF.P11

RUN-TIME: 11 15 .5 SECONDS

RUN-TIME RATIO: 248/27=9.1

CORE USED: 21K (41 PAGES)

DOCUMENT PAGES: 109