

# 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 frames with text, likely a table of contents or index. The remaining columns contain frames with data, possibly representing a list of items or a detailed table. The text is too small to be legible in this image.

B01

EOF1DZRPZCSEG

00010000 771114

POP10 411

ONHDR1DZRSBFSEG

00010000 771114

MAIN  
DEC-11  
DZRSB  
F11

.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

CONTENTS  
-----

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

PDP11 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	DRIVE 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. TEST
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. ↑U 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  
DZRSBF.P11 27-SEP-77 15:12

KEYING IN SWREG VALUE.



MAINDEC-11-DZRSB-F RH11-R503/LA-R504 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 SEQ 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 RH11-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, RSER, 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 RSMC SHOULD NOT MOVE, THIS IS ALSO TESTED.

19. ILLEGAL FUNCTION CODE TEST

IN THIS TEST RSBA, RSMC 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, RSMC 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 RH11-R503LA-R503-R504 BASIC FUNCTION DIAGNOSTIC MACY11 30(1046) 28-SEP-77 10:04 PAGE 51-1 SEG 0014  
DZRSB.F11 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 TRYS 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 R5BA. MOV -1000 INTO R5AC 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
. =	0	
. =	46	
000046	SENDAD	
. =	52	
000052	BIT14	
. =	174	
000174	DISPREG:0	:SOFTWARE SWITCK REGISTER LOCATION

```

000176 SWREG: 0
000200 . = 200
          JMP 2#STRT
000220 . = 220
          JMP 2#WRTLCK      :WRITE LOCK TEST
000230 . = 230
000236 STRT: BIC 2#BIT11 ONCEE :CLEAR TKSEL TEST FLAG
          JMP 2#BEGIN      :STARTING ADDRESS IS BEGIN

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

          . = 250
000250 JMP 2#SRSWC      :WORD COUNT REG
000254 JMP 2#SRSBA      :BUS ADDR REG
000260 JMP 2#SRSDA      :DISK ADDR REG
000264 JMP 2#SRSDS      :DRIVE STATUS REG
000270 JMP 2#SRSER      :ERROR REG
000274 JMP 2#SRSLA      :LOOK AHEAD REG
000300 JMP 2#SRCS2      :CS2 REG
000304 JMP 2#SRAS      :ATTENTION SUMMARY REG
000310 JMP 2#RSMRR      :MAINTENANCE REG
000314 JMP 2#SRSCS1     :CSI REG

          . = 320
000320 BIS 2#BIT11 ONCEE :SET TKSEL TEST FLAG
00032E JMP 2#BEGIN      :ENABLES OPERATOR TO DO A
          :WRITE OR READ TO A DESIRED ADDR
          :OPTIONAL WC, DA, AND UNIT #

```

# H02

```
N=          1
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=   2000
BIT11=   4000
BIT12=  10000
BIT13=  20000
BIT14=  40000
BIT15= 100000

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
001026 SWR:     177570
001030 DISPLAY: 177570

```

```

      . =      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 RSVCP5:   206     ; INTERRUPT PRIO. 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
KIPAR0=172340
KIPAR1=172342
KIPAR2=172344
KIPAR7=172356
KIPDR0=172300
KIPDR1=172302
KIPDR2=172304
KIPDR7=172316
RW=6
UP=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:  MLT !CS1,AS,BA
           MLT !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
SC=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: 0 ;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 WRNG 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 CLR LAD ;INIT LAD
001262 BIC #143777, ONCEE ;CLEAR ONCEE
001270 MOV #OUTBUF, @BUFSV ;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 1ST1: 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, @RS04T ; 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$: CMP #0, @RS04T ; IS THIS A RS03?
001476 BNE 2$ ; NO
001500 CLR RS04DT ; YES
001504 BR 1$ ; GET OUT
001506 2$: CMP #1, @RS04T ; IS THIS A RS03 4US?
001514 BNE 3$ ; NO
001516 CLR RS04DT ; RS03
001522 BR 1$ ; GET OUT
001524 3$: CMP #2, @RS04T ; IS THIS A RS04?
001532 BEQ 6$ ; YES
001534 CMP #3, @RS04T ; 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$: MOV UNNUM, -(6) ; PUT UNNUM ON STACK
001616 TYPES ; TYPE STACK IN OCTAL - SUPRESS
001620 TYPE ; TYPE SPACE
001624 BIC @BIT15, ONCEE ; CLEAR ERROR FLAG
001632 4$: BR NOWGO ; NOW TEST
    
```

```

001634 TRYNX: ASL UNCMP ;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,UNCMP ;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,JSRCS2 ;GET UNIT #
002070 MOV #177776,JSRCS1 ;SET ALL
002076 MOV #177777,JSRBA ;POSSIBLE R/W
002104 MOV #177777,JSRDA ;BITS IN THESE REGISTERS
002112 MOV #177777,JSRER
002120 MOV #177777,JSRMR
002126 MOV #177777,JSRWC
002134 MOV #177737,JSRCS2
002142 RESET ;CLEAR ALL BITS IN ALL REG.

```

:TEST RSCS2 FOR CLEARED BITS

```

002144 CMP #100,JSRCS2 ;DID THESE BITS GET CLEARED?
002152 BEQ +4 ;YES
002154 HLT !CS2 ;(417) SHOULD BE CLEARED IN CS2
002156 MOV UNNUM,JSRCS2 ;PUT # OF UNIT IN TEST IN CS2
002164 CMP #10600,JSRDS ;IS DPR AND MOL SET?
002172 BEQ +4 ;YES
002174 HLT !DS ;NO WHY NOT?

```

:TEST CONTROL AND STATUS REG 1

```

002176 CMP #4200,JSRCS1 ;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
:*****
002274 TST3: SCOPE

002276 TTAGG: MOV 8340,88PS :LOCK OUT INTERRUPTS
002304 MOV UNNUM,8RSCS2 :GET UNIT #
002313 MOV 843578,8RSCS2 :GET ALL
002320 MOV 817777,8RSCS1 :POSSIBLE
002326 MOV 817777,8RSCS1 :REGISTERS
002334 MOV 817777,8RSCS1
002342 MOV 817777,8RSCS1
002350 MOV 817777,8RSCS1
002356 MOV 820417,8RSCS2
002364 MOV 871,8RSMR
002372 MOV 840,8RSCS2 :CLEAR ALL BITS
002400 CMP 8100,8RSCS2 :DID THE RIGHT BITS CLEAR?
002406 BEQ +4 :YES
002410 HLT :CS2 (417) SHOULD BE CLEARED IN CS2
002412 MOV UNNUM,8RSCS2 :GET DRIVE NUMBER
002420 BIT 8173577,8RSCS1 :DID ALL BITS GET CLEARED
002426 BEQ +4 :YES
002430 HLT :CS1 :NO, ALL BITS SHOULD BE 0
:TEST BUS ADDRESS REGISTER

002432 TST 8RSCS1 :IS BA REG. CLEARED
002436 BEQ +4 :YES
002440 HLT :BA :SHOULD BE 0
:TEST DISK ADDRESS REGISTER

002442 TST 8RSCS1 :IS DA CLEARED
002446 BEQ +4 :YES
002450 HLT :BA :SHOULD BE 0
:TEST ERROR REG RSER

002452 TST 8RSCS1 :DID THESE BITS GET CLEARED
002456 BEQ +4 :YES
002460 HLT :RER :BITS(157015) SHOULD BE CLEARED
:TEST RS MAINTENANCE REGISTER

002462 BIT 877,8RSMR :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 8177777,8RSCS1 :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 CLRDK ;CLEAR ALL RS REG  
002512 MOV TIMSV,TIMES ;GET TIME  
002514 MOV #3576,RSCS1 ;SET DISK FUNCTION BITS  
002516 CMP #7776,RSCS1 ;ARE THESE BITS SET?  
002518 BEQ +4 ;NO  
002520 HLT !CS1 ;SHOULD = 7776  
002522 MOV #6724,RSCS1 ;SET THESE BITS  
002524 CMP #6724,RSCS1 ;DID THEY SET  
002526 BEQ +4 ;YES  
002528 HLT !CS1 ;SHOULD BE 6724  
002530 MOV #1052,RSCS1 ;SET THESE BITS  
002532 CMP #5252,RSCS1 ;ARE THEY =?  
002534 BEQ +4 ;YES  
002536 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 15 ;YES  
002662 MOV RSCS2,BAD  
002666 MOV #21137,GOOD ;WHAT CS2 SHOULD =  
002672 HLT ;CS2 = BAD GOOD = CORRECT ANS

```

002674 1S:  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      #17777, @RSBA   :SET THE BITS
002776      CMP      #17776, @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  1S:  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      1$                :LOOP TILL DONE

```



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.

003430 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 WD
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

```

003534 TST23: SCOPE

```

003536      MOV      #177017, @R5ER      :SET THESE BITS
003544      CLR      @R5ER                :CLEAR THEM
003550      TST      @R5ER                :DID THEY CLEAR
003554      BEQ      +4                    :YES
003556      HLT      !ER                  :SHOULD = 0
003560 TST24: SCOPE

```

:SET AND CLEAR RSMR

```

003562      MOV      #70, @R5MR          :SET THESE BITS
003570      MOV      @R5MR, 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, @R5MR          :SET BITS
003626      CLR      @R5MR                :CLEAR THEM
003632      BIT      #77, @R5MR          :DID THEY CLEAR
003640      BEQ      +4                    :YES
003642      HLT      !MR                  :BITS (77) SHOULD = 0
003644 TST26: SCOPE

```

```

003646      MOV      #50, @R5MR          :SET BITS
003654      MOV      @R5MR, 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, @R5MR          :SET BITS
003712      MOV      @R5MR, 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
*****
003740 TST30: SCOPE

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$: NCP
    
```

# 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           MOV            #5, @RSCS1B           ;LOAD BIT
004200           CMP            #6766, @RSCS1           ;DID IT LOAD?
004206           BEQ            +4                    ;YES
004210           HLT            !CS1
004212           MOV            #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           MOV            #377, @RSWCB           ;LOAD BIT
004314           CMP            #177652, @RSWC       ;DID IT LOAD?
004322           BEQ            +4                    ;YES
004324           HLT            !WC                   ;NO WC SHOULD =177652
004326           MOV            #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           MOV            #377, @RSBAB           ;LOAD BIT
004364           CMP            #177652, @RSBA       ;DID IT LOAD?
004372           BEQ            +4                    ;YES
004374           HLT            !BA                   ;DA SHOULD =177652
004376           MOV            #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 SILOB: CLRDK          :CLEAR ALL RS REG
004424      MOV      @RSDB,BAD      :READ FROM EMPTY SILO
004430      MOV      @RSCS2,BAD     :GET CS2
004434      MOV      #100100,GOOD   :GET CORRECT ANS
004440      BIS      UNNUM,GOOD     :FOR CS2
004444      CMP      BAD,GOOD       :IS CS2 CORRECT?
004446      BEQ      +4             :YES
004450      HLT      :CS2          :SHOULD HAVE DLT ERROR
004452      CMP      #144200,@RSCS1 :DID SC AND TRE SET?
004460      BEQ      +4             :YES
004462      HLT      :CS1          :SC AND TRE SHOULD BE SET
004464      MOV      @TRE,@RSCS1    :CLEAR ERROR BIT
004472      BIT      #140000,@RSCS1 :DID SC + TRE CLEAR
004500      BEQ      +4             :YES
004502      HLT      :CS1          :TRE AND SC SHOULD BE 0
004504      MOV      @RSCS2,BAD     :GET CS2
004510      BIC      #BIT15,GOOD    :GET CORRECT ANS
004514      CMP      GOOD,BAD      :IS CS2 CORRECT?
004516      BEQ      +4             :YES
004520      HLT      :CS2          :DLT SHOULD BE 0

```

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

004522

TST36: SCOPE

```

004524 ZERONE: CLRDK      :CLEAR ALL RS REG
004526      CLR      @RSDB         :LOAD DB WITH ALL 0
004532      MOV      #177777,@RSDB :LOAD DB WITH ALL ONES
004540      MOV      #2000,WORK     :TIME OUT ROUTINE
004546      MOV      #300,GOOD     :GET CORRECT FOR CS2
004552      BIS      UNNUM,GOOD
004556      2S:      MOV      @RSCS2,BAD :GET CS2
004562      CMP      GOOD,BAD      :IS IT CORRECT?
004564      BEQ      3S           :YES
004566      DEC      WORK         :TO WAIT FOR OR
004572      BNE      2S          :TO SET
004574      HLT      :CS2        :OR SHOULD BE SET
004576      3S:      CLR      GOOD
004600      MOV      @RSDB,BAD     :LOAD BAD WITH DB
004604      CMP      GOOD,BAD     :IS BAD CORRECT
004606      BEQ      +4           :YES
004610      HLT      :CS2        :COULD NOT FLOAT 0 THROUGH DB
004612      MOV      #-1,GOOD     :LOAD GOOD WITH ANS
004616      MOV      @RSDB,BAD   :GET DATA FROM DB
004622      CMP      GOOD,BAD    :IS DB CORRECT
004624      BEQ      +4           :YES
004626      HLT      :CS2        :BAD SHOULD = 177777

```

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

004632	SILO:	CLRDK		: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	:CS2	: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	:CS2	: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

MAINDEC-11-DZRSB-F

RH11-R503LA-R503-R504

BASIC FUNCTION DIAGNOSTIC MACY11 30(1046)

28-SEP-77

10:04

PAGE 71

SEQ 0037

DZRSBF.P11

27-SEP-77 15:12

TST36

LOAD RSDB WITH ALL ONES AND ALL ZEROS

:FLOAT A 1 AND A 0 THROUGH THE SILO

005004	SILOFL:	CLDK		: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,@RSCS1	:DO NO-OP FUNCTION
005152		BIT	@1,@RSCS1	:DID GO BIT CLEAR
005160		BEQ	:+4	:YES
005162		HLT	!CS1	: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:	CLDK	:	CLEAR ALL RS REG
005246		MOV	:	LOAD ER
005248		BIT	:	IS ATA BIT SET?
005250		BNE	:	YES
005252		HLT	:	AS BIT SHOULD BE SET
005254		MOV	:	DATA TO BE XFERED
005256		MOV	:	SET UP CURRENT ADDRESS
005258		MOV	:	LOAD MC WITH -1
005260		MOV	:	DO NO-OP FUNCTION
005262		BIT	:	DID GO BIT CLEAR
005264		BEG	:	YES
005266		HLT	:	GO BIT SHOULD BE CLEARED
005268		CMP	:	DID ERR BITS SET?
005270		BEG	:	NO
005272		HLT	:	ERR BIT SHOULD BE 0
005274		CMP	:	DID MC MOVE?
005276		BEG	:	NO
005278		HLT	:	MC SHOULD = 1777777
005280		TST	:	DID DA MOV
005282		BEG	:	NO
005284		HLT	:	DA SHOULD =0
005286		CMP	:	DID BA MOVE
005288		BEG	:	NO
005290		HLT	:	BA MOVED
005292		BIT	:	AS SHOULD BE SET
005294		BNE	:	IS IT?
005296		HLT	:	NO
005298		CMP	:	DID ER CHANGE?
005300		BEG	:	NO
005302		HLT	:	ER SHOULD NOT CHANGE

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

005422 TST43: SCOPE

005424	NOOP21: CLDK		:CLEAR ALL RS REG
005426	MOV	0177777,OUTBUF	:DATA TO BE XFERED
005434	MOV	000BUFSV,0RSBA	:SET UP CURRENT ADDRESS
005442	MOV	0-1,0RSWC	:LOAD MC WITH -1
005450	MOV	021,0RSCS1	:DO NO-OP FUNCTION
005458	BIT	01,0RSCS1	:DID GO BIT CLEAR
005466	BEG	:+4	:YES
005474	HLT	:CS1	:GO BIT SHOULD BE CLEARED
005482	TST	0RSER	:DID ANY ERRORS OCCUR?
005490	BEG	:+4	:NO
005498	HLT	:ER	:ALL ERROR BITS SHOULD BE 0
005506	CHP	0-1,0RSWC	:DID MC MOVE?
005514	BEG	:+4	:NO
005522	HLT	:MC	:MC SHOULD = 1777777
005530	TST	0RSDA	:DID DA MOV
005538	BEG	:+4	:NO
005546	HLT	:DA	:DA SHOULD =0
005554	CHP	000BUFSV,0RSBA	:DID BA MOVE
005562	BEG	:+4	:NO
005570	HLT	:BA	:BA MOVED
005578	BIT	UNCMP,0RSAS	:AS SHOULD NOT SET ON
005586	BEG	:+4	:A NO-OP FUNCTION
005594	HLT	:AS	:AS SET WHY?
005602	CHP	04220,0RSCS1	:IS CS1 CORRECT?
005610	BEG	:+4	:YES
005618	HLT	:DS	:CS1 SHOULD = 4220

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

005560 TST44: SCOPE

005562	NNOP21: CLRDK		:CLEAR ALL RS REG
005564	MOV	87,RSER	:LOAD ER
005572	BIT	UNCMP,RSAS	:IS ATA BIT SET?
005600	BNE	+4	:YES
005602	HLT	:AS	:AS BIT SHOULD BE SET
005604	MOV	8177777,OUTBUF	:DATA TO BE XFERED
005612	MOV	280BUFSV,RSBA	:SET UP CURRENT ADDRESS
005620	MOV	8-1,RSWC	:LOAD WC WITH -1
005626	MOV	821,RSCSI	:DO NO-OP FUNCTION
005634	BIT	81,RSCSI	:DID GO BIT CLEAR
005642	BEQ	+4	:YES
005644	HLT	:CSI	:GO BIT SHOULD BE CLEARED
005646	CMP	8150600,RSDS	:DID ERR BITS SET?
005654	BEQ	+4	:NO
005656	HLT	:DS	:ERR BIT SHOULD BE 0
005660	CMP	8-1,RSWC	:DID WC MOVE?
005666	BEQ	+4	:NO
005670	HLT	:WC	:WC SHOULD = 1777777
005672	TST	RSDA	:DID DA MOV
005676	BEQ	+4	:NO
005700	HLT	:DA	:DA SHOULD =0
005702	CMP	280BUFSV,RSBA	:DID BA MOVE
005710	BEQ	+4	:NO
005712	HLT	:BA	:BA MOVED
005714	BIT	UNCMP,RSAS	:AS SHOULD BE SET
005722	BNE	+4	:IS IT?
005724	HLT	:AS	:NO
005726	CMP	87,RSER	:DID ER CHANGE?
005734	BEQ	+4	:NO
005736	HLT	:ER	:ER SHOULD NOT CHANGE
005740	CMP	8104220,RSCSI	:IS CSI CORRECT?
005746	BEQ	+4	:YES
005750	HLT	:DS	:CSI 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

005754	DRCLR: CLRDK		: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	

```

: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,WAIRY      :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 HLT      !CS1!ER!DS!DA :ERROR DURING X-FER
006276 45: 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	RDTST:	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	1\$:	MOV	:	GO READ
006426	2\$:	TSTB	:	TEST FOR BUSY=1
006432		BPL	:	BUSY SET
006434		HLT	:	BUSY NOT SET
006436		JSR	:	WAIT FOR READY
006442		HLT	:	TIMEOUT RDY DID NOT SET
006444		CMP	:	WAS RSDA INCREMENTED BY 1
006452		BEQ	:	RSDA OK
006454		HLT	:	RSDA SHOULD CONTAIN A 1
006456	3\$:	CMP	:	IS ERROR FLAG SET?
006464		BEQ	:	NO! X-FER OK
006466		HLT	:	RSCS1 SHOULD = 270
006470	4\$:	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	RSWCWT:	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 28K. 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	WRTSTB: CLROK		: CLEAR ALL RS REG
006740	MOV	@OBUFSV, WORK	: GET LOC OF RSBA TO LOAD
006746	MOV	@177777, @WORK	: DATA TO BE X-FERED
006754	MOV	@OBUFSV, @RSBA	: SET UP CURRENT ADDRESS
006762	MOV	@-1, @RSWC	: SET WORD COUNT TO -1
006770	MOV	@2061, @RSCS1	: TEST B PORT
006776	2\$: TSTB	@RSCS1	: TEST FOR RDY=0
007002	BPL	+.4	: RDY=0
007004	HLT	:CS1	: RDY SHOULD = 0
007006	JSR	PC, WAITRY	: WAIT FOR READY
007012	HLT	:CS1	: SHOULD = 260 RDY NEVER CAME UP
007014	BIT	@BIT12, @SWR	: INHIBIT ADDRESS?
007022	BNE	3\$	: YES
007024	BIT	@BIT11, @RSCS2	: DID NEM SET?
007032	BEQ	3\$	: NO
007034	JMP	FINDM	: GO FIND MEMORY ON PORT B
007040	3\$: CMP	@1, @RSDA	: IS RSDA CORRECT
007046	BEQ	+.4	: RSDA OK
007050	HLT	:DA	: SHOULD = 1 SHOULD INCREMENT
007052	CMP	@6260, @RSCS1	: IS CS1 CORRECT?
007060	BEQ	4\$	: YES
007062	HLT	:CS1	
007064	4\$: TST	@RSWC	: FETCH WORD COUNT
007070	BEQ	+.4	: WORD COUNT DID OVERFLOW
007072	HLT	:WC	: SHOULD = 0 FAILED TO INCREMENT
007074	CMP	@10600, @RSDS	: IS RSDS OK?
007102	BEQ	+.4	: YES
007104	HLT	:DS!DA	: NO
007106	MOV	UNNUM, GOOD	: GET UNIT #
007112	BIS	@100, GOOD	: SET IR BIT
007116	MOV	@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	@RSBA, BAD	: GET BA DATA
007134	MOV	@OBUFSV, GOOD	: WHAT RSBA SHOULD EQUAL
007140	ADD	@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                ;CLR TO READ INTO
007162 MOV @#OBUFSV,@RSBA ;SET UP CURRENT ADDRESS
007170 MOV #-1,@RSWC        ;SET WORD COUNT TO -1
007176 MOV @2071,@RSCS1    ;B PORT
007204 2S: TSTB @RSCS1    ;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,@RSDA    ;WAS RSDA INCREMENTED BY 1
007230 BEQ .+4            ;RSDA OK
007232 HLT !DA!ER!DS     ;RSDA SHOULD CONTAIN A 1
007234 CMP @6270,@RSCS1  ;TST B PORT
007242 BEQ 4S            ;OK
007244 HLT !CS1          ;CS1 SHOULD = 6270
007246 4S: TST @RSWC     ;TEST WC
007252 BEQ .+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 @RSCS2,BAD    ;GET CS2
007272 CMP GOOD,BAD     ;IS CS2 CORRECT?
007274 BEQ .+4          ;YES
007276 HLT              ;GOOD = CORRECT ANS FOR CS2
007300 MOV @RSBA,BAD        ;FETCH CURRENT ADDRESS
007304 MOV @#OBUFSV,GOOD  ;WHAT RSBA SHOULD EQUAL
007310 ADD @2,GOOD        ;UPDATE IT
007314 CMP BAD,GOOD     ;IS RSBA CORRECT
007316 BEQ .+4          ;YES EXECUTE CONTINUE
007320 HLT              ;RSBA FAILED TO INCREMENT
007322 MOV @#OBUFSV,WORK  ;GET DATA READ FROM DISK
007330 MOV @WORK,BAD        ;
007334 MOV #-1,GOOD        ;GET CORRECT ANS
007340 CMP GOOD,BAD      ;IS OUTBUF CORRECT
007342 BEQ .+4          ;YES
007344 HLT              ;GOOD=CORRECT ANS BAD=DATA READ FROM DISK
    
```

\*\*\*\*\*  
:TEST 53 TEST WRITE 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	2S: 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	3S	:YES
007460	HLT	:CS1	:CS1 SHOULD = 6250
007462	3S: CMP	@BIT0,@RSDA	:WAS DAR INCREMENTED BY 1
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,OBUFSV ;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 3S ;NO THEN USE IT
007556 INC R4 ;GET WRONG DRIVE
007560 3S: MOV #177777,OUTBUF ;DATA TO BE X-FERED
007566 MOV #OBUFSV,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 1S: 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,RSDA ;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,RSRDS ;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 #OBUFSV,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		:CLEAR ALL RS REG
007756	MOV	@#OBUFSV,@RSBA	:SET UP CURRENT ADDR
007764	MOV	#-1,@R5WC	:SET WORD COUNT TO -1
007772	BIS	#BAI,@RSCS2	:SET BAI BIT
010000	MOV	#61,@RSCS1	:WRITE
010006	JSR	PC,WAITRY	:WAIT FOR READY
010012	HLT	!CS1	:RDY DID NOT SET
010014	IS: MOV	@#OBUFSV,GOOD	:WHAT RSBA SHOULD BE
010020	MOV	@RSBA,BAD	:WHAT RSBA IS
010024	CMP	GOOD,BAD	:COMPARE
010026	BEQ	.+4	:YES
010030	HLT		: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	4S: BIT	#1000, @RSMR	: WAIT FOR DISK TO
010244	BNE	3S	: REACH SECTOR 32
010246	DEC	WORK	: TIME OUT
010252	BNE	4S	: ROUTINE
010254	HLT	!MR	: COULD NOT FIND SECTOR 32

```

010256 38: CLR 2RSDA ;NOW SEARCH FOR 0
010258 MOV 831,2RSCS1 ;DO A BLOCK SEARCH FUNCTION
010270 BIT 8DRY,2RSDS ;IS DRY CLEARED?
010276 BEQ 18 ;YES
010300 HLT :DS ;DRY SHOULD BE CLEARED DURING A BLOCK SEARCH
010302 BR 0OUT ;GET OUT BECAUSE OF TIMING
010304 18: BIT 820000,2RSDS ;IS PIP SET?
010312 BNE +4 ;YES
010314 HLT :DS ;PIP SHOULD BE SET
010316 MOV 820000,GOOD ;SETUP FOR TIMEOUT ROUTINE
010322 25: DEC GOOD ;DO TIMEOUT
010324 BEQ TTOUT ;TIMED OUT
010326 BIT 8DRY,2RSDS ;DID DRY SET?
010334 BEQ 25 ;NO
010336 CMP 8110600,2RSDS ;DID PIP CLEAR?
010344 BEQ +4 ;YES
010346 HLT :DS ;PIP BIT DID NOT CLEAR
010350 CMP 8104230,2RSCS1 ;DID SC SET?
010356 BEQ +4 ;YES
010360 HLT :CS1:DS ;SC DID NOT SET
010362 MOV UNNUM,WORK ;GET CORRECT AS BIT
010370 CLR GOOD ;IN RSAS REG
010372 SEC ;THAT SHOULD
010374 55: ROL GOOD ;BE SET
010376 TST WORK
010402 BEQ 65
010404 DEC WORK
010410 BR 55
010412 65: CMP GOOD,2RSAS ;IS RSAS CORRECT?
010416 BEQ 75 ;YES
010420 MOV 2RSAS,BAD ;NO
010424 HLT
010426 75: MOV GOOD,2RSAS ;CLEAR AS REG
010432 TST 2RSAS ;DID IT CLEAR?
010436 BEQ +4 ;YES
010440 HLT :AS ;NO
010442 CMP 810600,2RSDS ;DID ATA CLEAR?
010450 BEQ +4 ;YES
010452 HLT :DS ;NO
010454 CMP 280BUFSV,2RSBA ;DID BA MOVE?
010462 BEQ +4 ;NO
010464 HLT :CS1:BA ;BA MOVED WHY?
010466 CMP 8-1,2RSWC ;DID WC MOVE?
010474 BEQ +4 ;NO
010476 HLT :WC ;WC MOVED WHY?
010500 BR 0OUT ;DONE GET OUT
010502 TTOUT: HLT :DS ;DYR NEVER CAME UP
010504 0OUT: ;DONE

```

\*\*\*\*\*  
: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,TIMSV      :SAVE LOOP COUNT
010514      MOV      B10,TIMES      :LOOP TEN TIMES
010522      CLRDK                    :CLEAR ALL RS REG
010524 18:  MOV      @ILLTAB,R3      :GET STARTING ADDR OF TABLE
010530 38:  MOV      (R3)+,BAD      :GET ILL FUN
010532      BEQ      ILFDN          :DONE GET OUT
010534      MOV      @B0B1FSV,@RSBA :SET UP REGS
010542      MOV      @-1,@RSMC      :TO CHECK FOR CHANGE
010550 28:  MOV      BAD,@RSCS1     :DO ILLEGAL FUNCTION
010554      BIC      @BIT0,BAD      :CLEAR GO BIT
010560      MOV      BAD,GOOD      :MOV ILLEGAL FUN INTO GOOD
010562 68:  TSTB      @RSCS1        :R0Y SET?
010566      BPL      @S                    :NO
010570      BIS      @104200,GOOD    :SET ERROR BITS
010574 48:  MOV      @RSCS1,BAD      :PUT CS1 INTO BAD
010600      CMP      GOOD,BAD      :IS CS1 CORRECT?
010602      BEQ      .+4           :YES
010604      HLT                    :GOOD IS WHAT CS1 SHOULD =BAD=CS1
010606      CMP      @1,@RSER      :DID ILF SET?
010614      BEQ      .+4           :YES
010616      HLT                    :ILF DID NOT SET
010620      CMP      @150600,@RSDS  :IS DS GOOD?
010626      BEQ      .+4           :YES
010630      HLT                    :ERR DID NOT SET
010632      MOV      @RSCS2,BAD      :GET CS2
010636      MOV      UNNUM,GOOD     :GET UNIT #
010642      BIS      @100,GOOD      :SET IR BIT
010646      CMP      GOOD,BAD      :IS CS2 CORRECT?
010650      BEQ      .+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,BAD      :GET RSAS REG
010670      CMP      BAD,GOOD      :DID CORRECT UNIT ANSWER?
010672      BEQ      .+4           :YES
010674      HLT                    :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      2-1,2RSWC   :DID WC MOVE?
010716 BEQ      +4              :NO
010720 HLT      :CS1:WC       :WC MOVED
010722 TST      2RSDA        :DID DA MOVE
010726 BEQ      +4              :NO
010730 HLT      :CS1:DA       :DA MOVED
010732 CLRDK          :CLEAR ALL ERRORS
010734 TST      2RSER        :DID ERRORS CLEAR
010740 BEQ      +4              :YES
010742 HLT      :DS          :ILF DID NOT CLEAR
010744 CMP      24200,2RSCS1 :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: CLRDK          :CLEAR ALL RS REG
010766 1$: MOV      2ILFTB2,R3 :GET TABLE OF ILL FUNS.
010772 3$: 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      2-1,2RSWC     :TO CHECK FOR CHANGE
011016 MOV      BAD,WORK1     :SHOULD WE TEST
011022 BIC      2177707,WORK1 :BA AND WC
011030 CMP      260,WORK1     :TO INC
011036 BNE      2$           :NO
011040 MOV      27,WORK        :YES
011046 2$: MOV      BAD,2RSCS1 :DO ILLEGAL FUNCTION
011052 10$: BIC     2BIT0,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 2144200,GOOD :SET ERROR BITS
011072 MOV 2RSCS1,BAD :PUT CS1 INTO BAD
011076 CMP GOOD,BAD :IS CS1 CORRECT?
011100 BEQ .+4 :YES
011102 HLT :GOOD IS WHAT CS1 SHOULD =BAD=CS1
011104 CMP 21,2RSER :DID ILF SET?
011112 BEQ .+4 :YES
011114 HLT :ILF DID NOT SET
011116 CMP 2150600,2RSDS :IS DS GOOD?
011124 BEQ .+4 :YES
011126 HLT :ERR DID NOT SET
011130 TST 2RSDA :DID DA MOVE?
011134 BEQ .+4 :NO
011136 HLT :CS1!DA :DA MOVED
011140 TST WORK :IS THIS AN ILL WRITE FUN?
011144 BNE 11S :YES
011146 MOV 2RSCS2,BAD :GET CS2
011152 MOV UNNUM,GOOD :GET UNIT #
011156 BIS 21100,GOOD :SET IR BIT
011162 CMP GOOD,BAD :IS CS2 CORRECT?
011164 BEQ .+4 :YES
011166 HLT :GOOD = CORRECT ANS FOR CS2
011170 CMP 2#0BUFSV,2RSBA :DID BA MOVE
011176 BEQ .+4 :NO
011200 HLT :CS1!BA :BA MOVED ON AN ILLEGAL FUNCTION
011206 CMP 2-1,2RSWC :DID WC MOVE?
011210 BEQ .+4 :NO
011212 HLT :CS1!WC :WC MOVED
011214 JMP 4S :CONTINUE UNTIL DONE
011220 11S: MOV 2RSCS2,BAD :GET CS2
011224 MOV UNNUM,GOOD :GET UNIT #
011230 BIS 21300,GOOD :SET IR BIT
011234 CMP GOOD,BAD :IS CS2 CORRECT?
011236 BEQ .+4 :YES
011240 HLT :GOOD = CORRECT ANS FOR CS2
011242 MOV 2#0BUFSV,WORK :GET BUFFER ADDR.
011250 ADD 2,WORK :UPDATE IT
011256 CMP WORK,2RSBA :DID BA MOVE
011264 BEQ .+4 :YES
011266 HLT :CS1!BA :BA MOVED ON AN ILLEGAL FUNCTION
011270 CMP 20,2RSWC :DID WC MOVE?
011276 BEQ .+4 :NO
011300 HLT :CS1!WC :WC MOVED
011302 4S: CLRDK :CLEAR ALL ERRORS
011304 CMP 24200,2RSCS1 :DID ERRORS CLEAR
011312 BEQ .+4 :YES
011314 HLT :DS :NO
011316 TST 2RSER :DID ERROR CLEAR
011322 BEQ .+4 :NO
011324 HLT :DS :YES
011326 JMP 3S :CONTINUE UNTIL DONE
011332 ILFDNE: :DONE WITH ILLEGAL FUNCTION TEST
    
```

F05

MAINDEC-11-DZRSB-F  
DZRSBF.P11

RH11-R503LA-R503-R504  
27-SEP-77 15:12

BASIC FUNCTION DIAGNOSTIC MACY11 30(1046)  
TST62 TEST ILLEGAL FUNCTION CODE 67

28-SEP-77 10:04 PAGE 90

SEG 0056

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

```

011332  TST62: SCOPE
011334  ILF67: CLDK
011336  MOV      # -1, @RSWC
011344  MOV      @#0BUF5V, WORK
011352  ADD      #2, WORK
011360  MOV      WORK, @RSBA
011366  MOV      #67, @RSCS1
011374  1S:    TSTB   @RSCS1
011400  BPL      1S
011402  MOV      @RSBA, BAD
011406  MOV      @#0BUF5V, GOOD
011412  CMP      GOOD, BAD
011414  BEQ      .+4
011416  HLT
011420  MOV      UNNUM, GOOD
011424  BIS      #1300, GOOD
011430  MOV      @RSCS2, BAD
011434  CMP      GOOD, BAD
011436  BEQ      .+4
011440  HLT
011442  CMP      #1, @RSER
011450  BEQ      .+4
011452  HLT
011454  CLDK
011456  TST      @RSER
011462  BEQ      .+4
011464  HLT
011466  MOV      # -100, @RSWC
011474  MOV      @#0BUF5V, WORK
011502  ADD      #200, WORK
011510  MOV      WORK, @RSBA
011516  MOV      #67, @RSCS1
011524  2S:    TSTB   @RSCS1
011530  BPL      2S

```

```

: CLEAR ALL RS REG
: SET WC TO -1
: GET OUTBUF ADD.
: FOR TEST
: LOAD ADDR
: DO FUNCTION 67
: DONE YET?
: NO
: GET BA REG
: GET CORRECT ANS FOR RSBA
: IS RSBA CORRECT?
: YES
: BAD=RSBA GOOD=CORRECT ANS.
: GET UNIT NUMBER
: SET IR AND OR BITS
: GET CS2
: IS CS2 CORRECT?
: YES
: BAD=CS2 GOOD=CORRECT ANS
: IS RSER CORRECT?
: YES
: ER IS WRONG
: CLEAR ALL RS REG
: DID ERROR CLEAR
: YES
: NO
: SET WC TO -100
: GET OUTBUF ADD.
: FOR TEST
: LOAD ADDR
: DO FUNCTION 67
: DONE YET?
: NO

```

```

011532 3S:  MOV      280BFSV,GOOD  :GET CORRECT ANS.
011534      MOV      2RSBA,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      21300,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      21,2RSER   :IS RSER CORRECT?
011600      BEQ      .+4         :YES
011602      HLT      :ER        :ER IS WRONG
011604      MOV      240011,2RSCS1 :CLEAR ERRORS
011612      CMP      24210,2RSCS1 :DID THEY CLEAR IN CS1
011620      BEQ      .+4         :YES
011622      HLT      :DS        :NO
011624      TST      2RSER      :DID RSER CLEAR
011630      BEQ      .+4         :YES
011632      HLT      :DS        :NO
011634      TST      2RSAS      :DID RSAS CLEAR
011640      BEQ      .+4         :YES
011642      HLT      :AS        :NO

```

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

```

011644 TST63: SCOPE
011646 PARTST: MOV      TIMSV,TIMES  :RESTORE LOOP #
011654      CLDK                     :CLEAR ALL RS REG
011656 1S:  MOV      210,2RSER      :SET PAR
011664      CMP      2150600,2RSDS  :DID ERR,ATA AND DRY SET?
011672      BEQ      .+4         :YES
011674      HLT      :DS!ER      :ER SHOULD SET IF PAR SETS IN RSER
011676      CLDK                     :CLEAR ALL RS REG
011700      TST      2RSER      :DID PAR CLEAR?
011704      BEQ      .+4         :YES
011706      HLT      :ER        :PAR DID NOT CLEAR BY CLEAR BIT
011710      CMP      210600,2RSDS  :DID ERROR BITS CLEAR
011716      BEQ      .+4         :YES
011720      HLT      :DS        :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,RSLA :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 #RSLA,GOOD :GET READING FROM LA  
011752 BIC #7700,GOOD :GET RID OF CS BITS  
011756 1\$: DEC WORK :WAIT FOR DISK  
011762 BEQ ERRR :TYPE ERROR  
011764 MOV #RSLA,BAD :READ LA  
011770 BIC #7700,BAD :CLEAR CS BITS  
011774 CMP GOOD,BAD :DID SF BITS CHANGE?  
011776 BEQ 1\$ :WAIT FOR TIME OUT  
012000 BR LATDON :LA OK CONT  
012002 ERRR: TYPE :.ASCIZ <15><12>"SECTOR FRACTIONS NOT MOVING"  
012002 HLT !LA :TYPE LOOK AHEAD REG  
012044 LATDON: :DONE CONT.  
012046

\*\*\*\*\*  
 :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 CLDK ;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 1$: MOV #-1,WORK ;SETUP TIMEOUT COUNTER
012134 INC RSDA ;GET NEXT SECTOR
012140 CMP #1000,RSDA ;DONE ALL YET?
012146 BEQ DONCS ;YES
012150 2$: DEC WORK ;DO TIMEOUT ROUTINE
012154 BEQ TMEOUT ;ADD. CON. NEVER CAME UP
012156 BIT #1000,RSMR ;DID ADD CONFIRM BIT SET?
012164 BEQ 2$ ;YES
012166 MOV RSLA,BAD ;GET LA
012172 BIC #77,BAD ;CLEAR SF BITS
012176 MOV #6,WORK1 ;SET UP COUNTER
012204 3$: ROR BAD ;MOV SA BITS RIGHT
012206 DEC WORK1 ;DO 6 TIMES
012212 BNE 3$ ;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 1$ ;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,2#4     ;SETUP TIME OUT VECTOR
012254      MOV    #340,2#6
012262      MOV    #MPRO,R2     ;GET PAR REG
012266 TSTAGN: TST    (R2)       ;DOES IT EXIST
012270      MOV    #WWP,2R2     ;YES SET WRITE WRONG PARITY
012274      MOV    #PARITY,R1   ;GET TEST LOCATION
012300 1S:   MOV    2R1,2R1     ;WRITE WRONG PARITY
012302      TST    2R1         ;READ IT
012304      TST    2R2         ;DID PARITY ERROR SET?
012306      BMI    2S         ;YES
012310      CLR    2R2         ;CLEAR PARITY REG
012312      BR    PRTP1       ;GET NEXT PARITY REG
012314 2S:   BIC    #100005,2R2 ;TURN OFF WWP,ENABLE BAD PARITY, ACTION ENABLE
012320      TST    2R2         ;PARITY ERROR?
012322      BEQ    NOPAR      ;IF NO BR.
012324      MOV    #1,2R2      ;CLEAR WWP AND ENABLE BAD PARITY AND ACTION ENABLE
    
```

```

012330      CLRDK
012332      MOV    #PARITY,2RSBA  ;CLEAR ALL RS REG
012340      MOV    #-1,2RSWC     ;SET UP CURRENT ADDRESS
012346      MOV    #61,2RSCS1   ;SET WORD COUNT TO -1
012354 3S:   TSTB   2RSCS1     ;GO WRITE
012360      BPL    3S         ;DONE YET?
012362      MOV    2RSCS2,BAD   ;NO WAIT
012366      MOV    #20100,GOOD  ;GET CS2
012372      BIS    UNNUM,GOOD   ;GET CORRECT AND FOR CS2
012376      CMP    GOOD,BAD    ;IS CS2 CORRECT?
012400      BEQ    .+4        ;YES
012402      HLT
012404      CMP    #144260,2RSCS1 ;CS2 SHOULD = GOOD
012412      BEQ    .+4        ;IS CS1 CORRECT?
012414      HLT              ;YES
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    #6,2#4
012450      CLR    2#6
012454      MOV    2R1,2R1     ;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:	CLDK	: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

```

012664 7S:  CMP      #144250, @RSCS1  :DID TRE SET?
012672      BEQ      .+4              :YES
012674      HLT      :CS1!ER!DS      :TRE SHOULD SET IF WCE SETS
012676      MOV      @RSBA, BAD        :FETCH CURRENT ADDRESS
012702      MOV      @@OBUF 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      :RSBA FAILED TO INCREMENT
012720      CLRDK   UNNUM, GOOD       :CLEAR ALL RS REG
012722      MOV      #100, GOOD       :PUT DRIVE IN GOOD
012726      BIS      @RSCS2, BAD      :SET IR BIT
012732      MOV      GOOD, BAD       :GET CS2
012736      CMP      .+4              :IS CS2 CORRECT
012740      BEQ      .+4              :YES
012742      HLT      :BAD =CS2 GOOD IS CORRECT ANS
012744      CMP      #4200, @RSCS1   :DID TRE CLEAR?
012752      BEQ      .+4              :YES
012754      HLT      :CS1           :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   #-1, OUTBUFF
013000      MOV      WCE↑
013006      JMP      :NOW WRITE -1 IN OUTBUF
013012  WCEDON: BIC      @BIT14, ONCEE :CLEAR TEST FLAG
    
```

```

:*****
: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,RSRCS1	: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	OBUSV,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		CLDK		:CLEAR ALL RS REG
013306		MOV	UNNUM,GOOD	:PUT DRIVE IN GOOD
013312		BIS	#100,GOOD	:SET IR BIT
013316		MOV	RSRCS2,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,RSRCS1	:DID TRE CLEAR?
013336		BEQ	.+4	:YES
013340		HLT	!CS1	:TRE DID NOT CLEAR WITH CLEAR
013342		BIT	#BIT14,ONCE	:FLOATION A 1 OR 0?
013350		BEQ	1\$	:FLOAT 1
013352		BR	6\$	:FLOAT 0
013354	WCEDNB:	BIS	#BIT14,ONCE	:SET BIT14
013362		CLDK		
013364		MOV	#-1,OBPORTT	
013372		JMP	WCETB	:NOW WRITE -1 IN OUTBUF
013376	WCEDOS:	MOV	OUTBUF,OBUSV	:RESTORE OBUSV
013404		BIC	#BIT14,ONCE	: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	@8081FSV,@RSDA	: SET UP CURRENT ADDRESS
013420	MOV	@177000,@RSMC	: SET WORD COUNT
013440	MOV	@61,@RSCS1	: GO WRITE
013444	2S: TSTB	@RSCS1	: IS RDY CLEARED YET?
013448	BMI	2S	: NO WAIT
013452	MOV	@71,@RSCS1	: GO READ
013456	JSR	PC,WAITRY	: WAIT FOR READY
013460	HLT	:CS1	: RDY NEVER CAME UP
013470	CMP	@144260,@RSCS1	: IS CS1 CORRECT?
013476	BEG	:+4	: YES
013500	HLT	:CS1	: TRE SHOULD SET BY SETTING PGE
013504	TST	@RSAS	: AS SHOULD = 0
013508	BEG	:+4	: YES
013510	HLT	:AS	: RSAS SHOULD = 0
013512	MOV	UNNUM,GOOD	: GET UNIT #
013516	BIS	@2300,GOOD	: SET PGE, IR, AND OR
013520	MOV	@RSCS2,BAD	: GET CS2
013524	CMP	GOOD,BAD	: IS IT CORRECT?
013530	BEG	:+4	: YES
013532	HLT		: BAD = CS2
013536	TST	@RSMC	: SHOULD NOT BE 0
013540	BNE	:+4	: BECAUSE PGE SHOULD ABORT
013542	HLT	:WC:CS1	: CURRENT OPERATION
013544	TST	@RSER	: DID ANY ERRORS SET?
013550	BEG	:+4	: NO
013552	HLT	:DS	: RMR SHOULD BE SET
013554	BIS	@TRE,@RSCS1	: CLEAR ERRORS
013562	BIC	@PGE,GOOD	: CLEAR PGE ERROR
013566	MOV	@RSCS2,BAD	: GET CS2
013572	CMP	GOOD,BAD	: IS CS2 CORRECT?
013574	BEG	:+4	: YES
013576	HLT	:CS2	: PGE DID NOT CLEAR BY CLEARING TRE BAD = CS2
013600	CMP	@4260,@RSCS1	: DID SC CLEAR
013606	BEG	:+4	: YES
013610	HLT	:DS	: DID NOT CLEAR BY CLEARING TRE

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

013612 TST73: SCOPE

013614	RMRT1:	CLRDK		:CLEAR ALL RS REG
013616		MOV	280BFSV,RSBA	:SET UP CURRENT ADDRESS
013624		MOV	8177700,RSWC	:SET WORD COUNT
013632		MOV	8172060,R3	:GET RSLA REG
013636	1S:	MOV	2A3,R4	:WAIT FOR
013640		BIC	817,R4	:THE MIDDLE
013644		CHP	820,R4	:OF SECTOR 0
013650		BNE	1S	:BEFORE DOING A SEARCH
013652		MOV	831,RSRCS1	:SEARCH
013660		MOV	87777,RSDA	:CAUSE ERROR
013666		JSR	PC,WAITRY	:WAIT FOR READY
013672		HLT	:CSI	:RDY NEVER CAME UP
013674		CHP	84,RSER	:DID RMR SET?
013702		BEQ	:+4	:YES
013704		HLT	:ER	:ER SHOULD = 4
013706		CHP	87777,RSDA	:DID DA GET MODIFIED?
013714		BNE	:+4	:NO
013716		HLT	:DA	
013720		CHP	8150600,RSDS	:DID ERR SET?
013726		BEQ	:+4	:YES
013730		HLT	:DS!ER	:ER DID NOT SET BECAUSE OF RMR
013732		CHP	8104230,RSRCS1	:IS CSI CORRECT?
013740		BEQ	:+4	:YES
013742		HLT	:DS	:CSI SHOULD = 144260
013744		CLRDK		:CLEAR ALL RS REG
013746		TST	RSER	:DID RMR CLEAR?
013752		BEQ	:+4	:YES
013754		HLT	:ER	:RMR DID NOT CLEAR WITH A CLEAR
013756		CHP	84200,RSRCS1	:IS CSI CORRECT?
013764		BEQ	:+4	:YES
013766		HLT	:DS	:NO

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

013770 TST74: SCOPE

013772	RMRT2:	CLRDK		:CLEAR ALL RS REG
013774		MOV	280BFSV,RSBA	:SET UP CURRENT ADDRESS
014002		MOV	8177700,RSWC	:SET WORD COUNT
014010		MOV	861,RSRCS1	:GO WRITE
014016	2S:	TSTB	RSRCS1	:IS RDY SET?
014022		BMI	2S	:YES WAIT FOR IT TO CLEAR
014024		MOV	8177773,RSER	:CAUSE ERROR

```

014032 JSR PC, WAITRY
014036 HLT !CSI :RDY NEVER CAME UP
014040 CMP #4, RSER :DID RMR SET?
014046 BEQ +4 :YES
014050 HLT !ER :ER SHOULD = 4
014052 CMP #150600, RSOS :DID ERR SET?
014060 BEQ +4 :YES
014062 HLT !DS!ER :ERR DID NOT SET BECAUSE OF RMR
014064 CMP #144260, RSOS1 :IS CSI CORRECT?
014072 BEQ +4 :YES
014074 HLT !DS :CSI SHOULD = 144260
014076 CLROK :CLEAR ALL RS REG

```

```

:*****
:TEST 75 TEST RMR IN RSER REGISTER TRYING TO WRITE INTO RSOS1
:*****

```

```

014100 TST75: SCOPE
014102 RMRT3: CLROK :CLEAR ALL RS REG
014104 MOV #80BUFSV, RSBA :SET UP CURRENT ADDRESS
014112 MOV #177700, RSMC :SET WORD COUNT
014120 MOV #61, RSOS1 :GO WRITE
014126 ZS: TSTB RSOS1 :IS RDY SET?
014132 BMI ZS :YES WAIT FOR IT TO CLEAR
014134 MOV #30, RSOS1 :CAUSE ERROR
014142 JSR PC, WAITRY :WAIT FOR READY
014146 HLT !CSI :RDY NEVER CAME UP
014150 CMP #4, RSER :DID RMR SET?
014156 BEQ +4 :YES
014160 HLT !ER :ER SHOULD = 4
014162 CMP #150600, RSOS :DID ERR SET?
014170 BEQ +4 :YES
014172 HLT !DS!ER :ERR DID NOT SET BECAUSE OF RMR
014174 CMP #144260, RSOS1 :IS CSI CORRECT?
014202 BEQ +4 :YES
014204 HLT !DS :CSI SHOULD = 144260
014206 CLROK :CLEAR ALL RS REG

```

E06

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

014210 TST76: SCOPE

014212	RMRT4:	CLROK		:CLEAR ALL RS REG
014214		MOV	#0BUFSV,RSBA	:SET UP CURRENT ADDRESS
014222		MOV	#177700,RSWC	:SET WORD COUNT
014230		MOV	#172060,R3	:GET RSLA REG
014234	1\$:	MOV	R3,R4	:WAIT FOR
014236		BIC	#17,R4	:THE MIDDLE
014242		CMP	#20,R4	:OF SECTOR 0
014246		BNE	1\$	:BEFORE DOING A SEARCH
014250		MOV	#31,RS1	:SEARCH
014256		MOV	#0,RSAS	:TRY TO CAUSE ERROR
014264		CLR	WORK	:CLEAR COUNTER
014270	2\$:	BIT	#BIT7,RS1	:WAIT FOR DRY
014276		BNE	3\$	:READY CONT
014300		INC	WORK	:COUNT
014304		BNE	2\$	:RETRY
014306		HLT	!CS1	:RDY NEVER CAME UP
014310	3\$:	TST	RSER	:DID RMR SET?
014314		BEQ	+4	:NO
014316		HLT	!ER	:ER SHOULD = 0
014320		CMP	#110600,RS1	:DID ERR SET?
014326		BEQ	+4	:NO
014330		HLT	!DS!ER	:DS SHOULD = 110600
014332		CMP	#104230,RS1	:IS CS1 CORRECT?
014340		BEQ	+4	:YES
014342		HLT	!DS	:CS1 SHOULD = 144260
014344		CLROK		:CLEAR ALL RS REG
014346		CMP	#4200,RS1	:IS CS1 CORRECT?
014354		BEQ	+4	:YES
014356		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$:      MOV          WORK2,RSWC   :LOAD WC
014434      MOV          #-1,OUTBUF     :WRITE -1
014442      MOV          @80BUFV,RSBA   :SET UP CURRENT ADDRESS
014450 4$:      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
014472 2$:      CLR          RS0DA      :SET DSK ADDRESS TO 0
014476      CLR          OUTBUF        :WRITE 0
014502      MOV          @80BUFV,RSBA   :SET UP CURRENT ADDRESS
014510      MOV          #-1,RSWC      :LOAD WC
014516      MOV          #172060,R2     :PUT RSLA ADDR INTO R2
014522 3$:      MOV          (R2),R3    :GET LA AND WAIT FOR
014524      BIC          #77,R3        :SECTOR 40
014530      CMP          #4000,R3      :BEFORE
014534      BNE          3$              :WRITING
014536      MOV          #61,RS0C1     :GO WRITE
014544 6$:      MOV          (R2),R3    :GET RSLA AND WAIT FOR
014546      BIC          #17,R3        :MIDDLE OF SECTOR
014552      CMP          #20,R3        :0 BEFORE EXECUTING
014556      BNE          6$              :A CLEAR FUNCTION
014560      MOV          #40,RS0C2     :CLEAR ALL REG. DO IT THIS WAY
014566      MOV          UNNUM,RS0C2   :DO NOT USE TRAP
    
```

```

014574 INCH: TSTB 2RSCS1      : IS BUSY CLEARED
014600 BAI 5$          : FLAG CLEARED
014602 HLT !CS1        : RDY NEVER CAME UP
014604 6$: MOV 280BUFSV,2RSBA : SET UP CURRENT ADDRESS
014612 MOV WORK2,2RSWC    : LOAD WC
014620 MOV 871,2RSCS1    : GO READ
014626 5$: TSTB 2RSCS1  : IS RDY SET?
014632 BPL 5$          : 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 CS1 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 1$: MOV 2RSWC,BAD : DID TRANSFER STOP AT END OF SECTOR?
014724 CMP GOOD,BAD
014726 BEQ +4          : YES
014730 HLT
014732 MOV 8OUTBUF,GOOD   : GET BA
014736 CMP 84,RS04DT    : LA DISK?
014744 BNE 7$          : NO
014746 ADD 8100,GOOD    : YES
014752 BR 3$          : CONTINUE
014754 7$: TST RS04DT  : RS04?
014760 BNE 2$          : YES
014762 ADD 8200,GOOD    : RS03
014766 BR 3$
014770 2$: ADD 8400,GOOD : GET CORRECT ANS FOR BA
014774 3$: MOV 2RSBA,BAD : GET BA
015000 CMP BAD,GOOD      : IS BA CORRECT?
015002 BEQ +4          : YES
015004 HLT
015006 CLRDK          : NO
015010 TST 2RSER        : CLEAR ALL RS REG
015014 BEQ +4          : DID DCK CLEAR?
015016 HLT !ER        : YES
015020 MOV 8177500,2RSWC   : DCK DID NOT CLEAR WITH CLEAR
015026 MOV 280BUFSV,2RSBA : CLEAR DCK ON
015034 MOV 861,2RSCS1    : DRIVE BY WRITING
015042 4$: TSTB 2RSCS1 : GOOD DATA
015046 BPL 4$          : ON DRIVE
    
```

: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		BEQ	:+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		BEQ	:+4	:YES
015200		HLT	:ER	:IAE SHOULD BE SET
015202		CMP	#150600, @RSDS	:DID ERR SET?
015210		BEQ	:+4	:YES
015212		HLT	:DS!AS	:ERR SHOULD BE SET
015214		CMP	#144260, @RSCS1	:DID SC SET?
015222		BEQ	:+4	:YES
015224		HLT	:CS1	:SC SHOULD BE SET
015226		CLRDK		:CLEAR ALL RS REG
015230		TST	@RSER	:CLR ERRORS?
015234		BEQ	:+4	:YES
015236		HLT	:ER	:ERR DID NOT CLR WITH 40 IN CS2

: 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: CLRDK		: 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, ARSCS2	: LOAD NED IN CS2
015302	CLR	ARSDA	: WRITE DRIVE REG
015306	TST	ARSER	: DID ANY BITS SET IN RSER?
015312	BEQ	.+4	: NO
015314	HLT	:DS	: WHY DID RSER CHANGE?
015316	MOV	ARSCS2, 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, ARSCS1	: IS CS1 CORRECT?
015342	BEQ	.+4	: YES
015344	HLT	:CS2	: TRE SHOULD SET BY NED ERROR
015346	TST	ARAS	: DID ANY BITS SET?
015352	BEQ	.+4	: NO
015354	HLT	:AS	: WHY DID AT BITS SET?
015356	MOVB	#100, ARSCS1B	: CLEAR TRE
015364	BIT	#NED, ARSCS2	: DID NED CLEAR
015372	BEQ	.+4	: YES
015374	HLT	:CS2	: NED DID NOT CLEAR
015376	MOV	ARSDA, WORK	: READ DRIVE REG
015404	BIT	#NED, ARSCS2	: 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"
015502	NNDD: BIS	#BIT12.ONCEE	: SET TYPED FLAG

```

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

```

015510
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      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$: MOV          #7777,RSDA :SET RSDA=TO ALL ONES
015564      2$: 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,@RSDA	:SET RSDA=TO ALL ONES
015742	2\$:	MOV	@0BUF5V,@RSBA	:CURRENT ADDRESS=OUTBUF
015750		MOV	#61,@RSCS1	:WRITE
015756		JSR	PC,WAITRY	:WAIT FOR READY
015762		HLT	:CS1	:RDY DID NOT SET
015764		TST	@RSEB	:ANY ERRORS?
015770		BEQ	:+4	:NO
015772		HLT	:ER	:YES
015774		CMP	#12600,@RSDS	:DID LBT SET?
016002		BEQ	:+4	:YES
016004		HLT	:DS	:LBT DID NOT SET
016006		TST	@RSCS1	:IS ERROR FLAG SET
016012		BPL	:+4	:NO
016014		HLT	:CS1	:ERROR
016016		CLRDK		:CLEAR ALL RS REG
016020		CMP	#10600,@RSDS	: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: CLRDK		:CLEAR ALL RS REG
016036	MOV	#177017, ARSER	:LOAD ER
016044	MOV	ARAS, 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, ARSCS1	:DID ERAS SET IN CS1?
016074	BEQ	:+4	:YES
016076	HLT	:DS	:CS1 SHOULD =104200
016100	MOV	UNCMP, ARAS	:CLEAR ATA BIT
016106	TST	ARAS	:DID IT CLEAR?
016112	BEQ	:+4	:YES
016114	HLT	:AS	:COULD NOT CLEAR AS BIT
			:BY LOADING A 1 INTO IT
016116	CMP	#4200, ARSCS1	:DID SC CLEAR BY
016124	BEQ	:+4	:CLEARING ATA
016126	HLT	:ER	:NO
016130	MOV	#177777, OUTBUF	:DATA TO BE XFERED
016136	MOV	ARBUFSV, ARSBA	:SET UP CURRENT ADDRESS
016144	MOV	#-1, ARSWC	:LOAD WC WITH -1
016152	MOV	#71, ARSCS1	:DO READ FUNCTION
016160	BIT	#1, ARSCS1	:DID GO BIT CLEAR
016166	BEQ	:+4	:YES
016170	HLT	:CS1	:GO BIT SHOULD BE CLEARED
016172	1\$: TSTB	ARSCS1	:WAIT FOR READY
016176	BPL	1\$	:WAIT
016200	CMP	#144270, ARSCS1	:DID ERAS CLEAR BY SETTING GO BIT?
016206	BEQ	:+4	:YES
016210	HLT	:ER	:NO

M06

016212	MOV	RS2, 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, RS2S	: 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	RS2A	: DID DA MOV
016264	BEQ	. +4	: NO
016266	HLT	: DA	: DA SHOULD = 0
016270	CMP	#0BUFSV, RS2B	: DID BA MOVE
016276	BEQ	. +4	: NO
016300	HLT	: BA	: BA MOVED
016302	BIT	UNCOMP, RS2S	: AS SHOULD BE SET
016310	BNE	. +4	: IS IT?
016312	HLT	: AS	: NO
016314	CMP	#177017, RSR	: 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: CLDK		:CLEAR ALL RS REG
016332	BIS	#BIT4,RSRCS2	:SET PAT
016340	TST	RSRMR	:READ DRIVE REG
016344	MOV	RSRCS2,BAD	:GET CS2
016350	MOV	#120,GOOD	:MCPE SHOULD
016354	BIS	UNNUM,GOOD	:NOT SET
016360	CMF	GOOD,BAD	:IS CS2 CORRECT?
016362	BEQ	.+4	:YES
016364	HLT		:BAD = CS2 GOOD = CORRECT ANS
016366	MOV	#10,RSRMR	:CAUSE PAR TO SET IN RSER
016374	CMP	#10,RSRER	: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,RSRCS1	:DID ERRS SET IN CS1?
016436	BEQ	.+4	:YES
016440	HLT	:DS	:CS1 SHOULD =104200
016442	CLDK		:CLEAR RS REG
016444	CMP	#4200,RSRCS1	:IS CS1 CORRECT?
016452	BEQ	.+4	:CLEARING ATA
016454	HLT	:ER	:NO
016456	MOV	RSRCS2,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: CLDK
016502      BIS      @BA1,@RSCS2      ;CLEAR ALL REG
016504      MOV      @177777,OUTBUF   ;SET BA1
016512      MOV      @808BUFSV,@RSBA  ;DATA TO BE XFERED
016520      MOV      @-1000,@RSMC    ;SET UP CURRENT ADDRESS
016526      BIS      @BIT4,@RSCS2    ;LOAD MC WITH -1
016534      MOV      @71,@RSCS1     ;SET PAT BIT
016542      TSTB    @RSCS1          ;DO READ FUNCTION
016550      BPL     18              ;WAIT FOR READY
016554      CMP     @144270,@RSCS1   ;WAIT
016556      BEQ     +4              ;DID CS1 GET LOADED?
016564      HLT     :DS             ;NO
016566      CMP     @808BUFSV,@RSBA  ;IT SHOULD NOT
016570      BEQ     +4              ;DID BA MOVE?
016576      HLT     :BA            ;NO
016600      CMP     @-1000,@RSMC     ;YES
016602      BEQ     +4              ;DID MC MOVE?
016610      HLT     :MC            ;NO
016612      HLT     :MC            ;YES WHY?
    
```

```

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

```

016614 FUNDO: CLDK
016616      BIS      @BA1,@RSCS2      ;CLEAR ALL REG
016620      MOV      @177777,OUTBUF   ;SET BA1
016626      MOV      @808BUFSV,@RSBA  ;DATA TO BE XFERED
016634      MOV      @-1000,@RSMC    ;SET UP CURRENT ADDRESS
016642      MOV      @71,@RSCS1     ;LOAD MC WITH -1
016650      BMI     38              ;DO A READ
016656      BIS      @BIT4,@RSCS2    ;WAIT FOR BUSY
016660      TSTB    @RSCS1          ;SET PAT
016666      BPL     28              ;WAIT FOR READY
016672      CMP     @144270,@RSCS1   ;DID MCPE SET?
016674      BEQ     +4              ;NO
016702      HLT     :ER            ;YES
016704      MOV      @RSCS2,BAD     ;GET CS2
016706      MOV      @730,GOOD     ;GET CORRECT ANS
016712      BIS      UNNUM,GOOD     ;GET UNIT #
016716      CMP     GOOD,BAD        ;IS CS2 CORRECT?
016722      BEQ     +4              ;YES
016724      HLT     :GOOD          ;GOOD = WHAT CS2 SHOULD =
016726
    
```

```

016730 CMP #10600,RSDS :DID ERR BITS SET?
016736 BEQ +4 :NO
016740 HLT :DS :ERR BIT SHOULD BE 1
016742 TST RSRER :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,RS1 :IS CS1 CORRECT?
016762 BEQ +4 :YES
016764 HLT :DS

```

```

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

```

016766 TST111: SCOPE

```

016770 PATTST: CLROK :CLEAR ALL RS REG
016772 BIS #BA1,RS2 :SET BAI
017000 MOV #17777,OUTBUF :DATA TO BE XFERED
017006 MOV #080BFSV,RSBA :SET UP CURRENT ADDRESS
017014 MOV #1000,RSWC :LOAD WC WITH -1
017022 MOV #61,RS1 :DO WRITE FUNCTION
017030 1S: TSTB RS1 :WAIT FOR READY
017034 BIS 1S :WAIT
017036 2S: BIS #BIT4,RS2 :SET PAT
017044 TSTB RS1 :WAIT FOR READY
017050 BPL 2S
017052 CMP #144260,RS1 :DID MCPE SET?
017060 BEQ +4 :NO
017062 HLT :ER :YES
017064 MOV RS2,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,RSER :DID PAR SET?
017132 BEQ +4 :YES
017134 HLT :DS!DA :NO
017136 CLROK :CLEAR ALL RS REG
017140 CMP #4200,RS1 :IS CS1 CORRECT?
017146 BEQ +4 :YES
017150 HLT :DS
017152 TST RSRER :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: CLDK		:CLEAR ALL RS REG
017166	MOV	#7777,RSDA	:SET RSDA=TO ALL ONES
017174	CMP	#4,RS04DT	:LA DISK?
017202	BNE	2S	:NO
017204	MOV	#-40,RSWC	:LOAD WORD COUNT
017212	BR	1S	:CONTINUE
017214	2S: MOV	#-100,RSWC	:WORD COUNT=-100
017222	TST	RS04DT	:IS THIS A RS04?
017226	BEQ	1S	:NO
017230	MOV	#-200,RSWC	:YES
017236	1S: MOV	#0BUF5V,RSBA	:CURRENT ADDRESS=OUTBUF
017244	MOV	#61,RS0CS1	:WRITE
017252	JSR	PC,WAIRY	: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,RS0DS	: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	CLDK		:CLEAR ALL RS REG
017314	CMP	#10600,RS0DS	: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: CLDK
017332 MOV # -1,OUTBUF ;CLEAR ALL RS REG
017340 MOV @#0BUFV,RSBA ;PUT - INTO OUTBUF
017346 CMP #4,RS04D ;SET UP CURRENT ADDR
017354 BNE 4$ ;RS03LA DISK?
017356 MOV # -40,RSWC ;LOAD WORD COUNT
017364 BR 5$ ;CONTINUE
017366 4$: MOV # -200,RSWC ;LOAD WC FOR RS04
017374 TST RS04D ;RS04?
017400 BNE 5$ ;YES
017402 MOV # -100,RSWC ;SET WORD COUNT TO -100
017410 5$: BIS #BAI,RS0CS2 ;SET BAI BIT
017416 MOV #61,RS0CS1 ;WRITE
017424 3$: TSTB RS0CS1 ;IS RDY SET?
017430 BPL 3$ ;NO
017432 CLR @RSDA ;SET DSK ADDRESS TO 0
017436 MOV # -1,OUTBUF ;PUT 177777 INTO OUTBUF
017444 MOV @#0BUFV,RSBA ;SET UP CURRENT ADDR
017452 MOV # -1,RSWC ;SET WORD COUNT TO -1
017460 BIS #10,RS0CS2 ;SET BAI BIT
017466 MOV #61,RS0CS1 ;WRITE
017474 1$: TSTB RS0CS1 ;IS RDY SET?
017500 BPL 1$ ;NO
017502 BIS #10,RS0CS2 ;CLEAR BAI BIT
017510 TST RS04D ;RS04?
017514 BEQ 7$ ;NO
017516 MOV #200,WORK ;YES
017524 BR 8$ ;CONT
017526 7$: MOV #100,WORK ;SET UP BUFFER
017534 8$: MOV @#0BUFV,R1 ;GET STARTING ADD OF BUF
017540 MOV # -1,(R1)+ ;LOAD FIRST WD WITH -1
017544 6$: CLR (R1)+ ;LOAD REST WITH 0
017546 DEC WORK ;DONE YET?
017552 BNE 6$ ;NO
017554 CLR @RSDA ;SET DSK ADDRESS TO 0
017560 MOV @#0BUFV,RSBA ;SET UP CURRENT ADDR
017566 CMP #4,RS04D ;RS03LA DISK?
017574 BNE 11$ ;NO
017576 MOV # -40,RSWC ;LOAD WORD COUNT
017604 BR 10$ ;CONTINUE
017606 11$: TST RS04D ;RS04?
017612 BEQ 9$ ;NO
017614 MOV # -200,RSWC ;YES
017622 BR 10$ ;CONT
017624 9$: MOV # -100,RSWC ;SET WORD COUNT TO -100
017632 10$: MOV #51,RS0CS1 ;WRITE CHECK

```

F07

```

017640 2S: BIT      #200,2RSCS1      : IS RDY SET?
017646      BEQ      2S              : 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#KIPDR0 : SET KIPDR0=RW UP 400 BLOCKS
017774      MOV      #400*256.-400+UP+RW,2#KIPDR1 : SET KIPDR1=RW UP 400 BLOCKS
020002      MOV      #400*256.-400+UP+RW,2#KIPDR2 : SET KIPDR2=RW UP 400 BLOCKS
020010      MOV      #400*256.-400+UP+RW,2#KIPDR7 : SET KIPDR7=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,RS1             :WRITE TWO WORDS ON DISK, RSBA
                                       :STARTS AT 177777 TO FORCE CARRY
                                       :TO SET A16
                                       :WAIT FOR READY

020056  TSTB     RS1               :
020062  BPL      -4                :
020064  TST     RS1               :
020070  BPL      1$                :
020072  HLT     !ER!DA!DS         :STATUS ERROR AFTER 2 WORD WRITE
020074  BR      2$                :USING MEXO
020076 1$: CMP     #4660,RS1       :IS CS1 CORRECT
020104  BEQ     3$                :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,RS1           :READ TWO WORDS INTO LOCATIONS
                                       :177777 AND 200000.
                                       :WAIT FOR READY

020142  TSTB     RS1               :
020146  BPL      -4                :
020150  TST     RS1               :ANY ERRORS?
020154  BPL      4$                :BRANCH IF NO
020156  HLT     !ER               :ERROR OFTER READING 2 WORDS
020160  BR      2$                :
020162 4$: CMP     #4670,RS1       :IS CS1 CORRECT?
020170  BEQ     5$                :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     :DATA COMPARE ERROR AT 200000
020214 2$: NOP

```

# H07

```

020216    EXTT1:  CLDK                                    ;CLEAR ALL REG
020220            MOV            #4000, @KIPAR2                   
020226            MOV            #40000, R2                    ;R2 EQUALS THE BASE ADDR
020232            7$:    MOV            #177777, (R2)               ;INSERT PATTERN INTO 400000
020236            MOV            #177777, @RSBA                ;SETUP BUS ADDR
020244            MOV            #-2, @RSWC                    ;LOAD WC
020252            MOV            #461, @RSCS1                ;SET BIT A16 AND WRITE
020260            TSTB            @RSCS1                    ;WAIT FOR READY
020264            BPL            -4                           
020266            TST            @RSCS1                    ;ANY ERRORS?
020272            BPL            4$                            ;BRANCH IF NO
020274            HLT            !ER                        ;ERROR AFTER READING 2 WORDS
020276            4$:    CMP            #5260, @RSCS1            ;IS CS1 CORRECT?
020304            BEQ            10$                        ;BRANCH IF YES
020306            HLT            !ER                        ;NO CS1 DID NOT COMPARE
                                  ;READ STARTING AT 377777
020310    10$:    CLR            (R2)                        ;CLEAR LOCATION 400000
                                  ;READ TWO WORDS STARTING AT 377777
020312            MOV            #-2, @RSWC                    ;SETUP WC
020320            CLR            @RSDA                        ;SETUP DA
020324            MOV            #177777, @RSBA               
020332            MOV            #471, @RSCS1                ;CLEAR A 17 SET A16, READ
020340            TSTB            @RSCS1                    ;WAIT FOR READY
020344            BPL            -4                           
020346            TST            @RSCS1                    ;ANY ERRORS?
020352            BPL            11$                        ;BRANCH IF NO
020354            HLT            !ER                        ;ERROR WHILE READING TWO WORDS
020356            BR            EXTRP                       
020360            11$:    CMP            #5270, @RSCS1            ;IS CS1 CORRECT?
020366            BEQ            12$                        ;BRANCH IF YES
020370            HLT            !ER                        ;CS1 DID NOT COMPARE
                                  ;READ STARTING AT 377777
020372            12$:    CMP            #177777, (R2)            ;WAS DATA READ INTO LOCATION 400000
020376            BEQ            EXTRP                        ;CORRECTLY? - BRANCH IF YES
020400            MOV            #177777, GOOD               
020404            MOV            (R2), BAD                   
020406            HLT                                        ;DATA COMPARE ERROR AT 400000 IF
                                  ;RECEIVED=0 - LOCATION WASN'T ACCESSED
                                  ;TURN OFF MEMORY MANAGEMENT
020410    EXTRP:  CLR            @RSRO                       
020414            BR            EXT1                       
020416    EXTTRP: NOP                                       
020420    EXT1:    MOV            #500, SP                    ;UPDATE TEST NUMBERS
020424    MEMOUT: MOV            #6, 4                        ;RESTORE STACK
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,ARSVEC :SET UP VECTOR
020454      MOV      #340,ARSVCPs :SET TRAP PS
020462      MOV      #200,APS     :SET PS AT PRIORITY 4
020470      MOV      #300,ARSCS1  :THIS SHOULD CAUSE A TRAP
020476      MOV      #500,WORK    :SETUP LOOP
020504 1$: DEC      WORK         :DEC LOOP SHOULD
020510      BNE     1$          :INTERRUPE 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,ARSCS1  :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,ARSVEC :SET UP INTERRUPT VECTOR
020560      MOV      #340,ARSVCPs :SET PRIO.
020566      MOV      #240,APS     :LOCK OUT ALL INTERRUPTS ABOVE
020574      MOV      APS,BAD     :GET PS
020600      MOV      #177777,ARSWC :SET WORD COUNT TO -1
020606      MOV      #0BUFSV,ARsBA :LOAD CURRENT ADDRESS
020614      MOV      #161,ARSCS1  :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:          :DONE GET OUT
    
```

```

:*****
: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      #0BUF5V,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     !ER!DS       ;GOOD=CORRECT PS BAD=WRONG PS
020740          HLT     !ER!DS
020742          BR      DONINT        ;CONT
020744 INT114:  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      #0BUF5V,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  
\*\*\*\*\*

021064

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

021066

MODNUM: CLDK :CLEAR ALL RS REG

021070

MOV #10, TIMES :SAVE LOOP COUNT

021076

MOV WORK1 :LOOP ONLY 10 TIMES

021104

CLR R3 :CLEAR WORK LOC.

021110

CLR R4

021112

MOV #DVTAB, R2 :SETUP TABLE

021114

MOV #401, WORK :SETUP TO TEST FOR MORE DRIVES

021120

7S:

BIT WORK, UNITSV :IS DRIVE ON SYSTEM?

021126

BEG 6S :NO

021134

CMP R4, UNNUM :IS THIS THE SAME DRIVE?

021136

BNE 8S :NO

021142

6S:

INC R4 :UPDATE 2ND UNIT #

021144

CLC

021146

ROL WORK :CHECK FOR NEXT DRIVE

021150

BCC 7S :NOT DONE YET

021154

BIT #BIT3, WORK1 :MULTI DRIVE?

021156

BNE 12S :YES

021164

MOV UNNUM, RS :LOAD UNIT NO

021166

INC RS :CHANGE IT

021172

BIC #177770, RS :CLEAR JUNK

021174

BR 12S

021200

8S:

BIS #BIT3, WORK1 :SET FOUND MULTI DRIVE

021202

MOV R4, (R2)+ :LOAD UNT # INTO TABLE

021210

INC R3 :COUNT # OF DRIVES

021212

MOV R3, SAVEE :SAVE IT

021214

BR 6S

021220

12S:

MOV #TSTVEC, JRSVEC :SETUP INT. TRAP

021222

MOV #340, JRSVCPS :SETUP PRIO.

021230

CLR OUTBUF :CLR TO READ INTO

021236

MOV #OBUFSV, JRSBA :SET UP CURRENT ADDRESS

021242

MOV #-1000, JRSWC :SET WORD COUNT

021250

1S:

MOV #60, JRSDA :LOAD DA

021256

MOV #DVTAB, R2 :GET TABLE

021264

MOV #161, JRSCS1 :GO WRITE

021270

TST R3 :MORE THEN 1 DRIVE?

021276

BNE 13S :YES

021300

MOV R5, JRSCS2 :NO MODIFY UNIT #

021302

BR 14S

021306

```

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 ;CSI SHOULD = 14270 OR 4270
021366 HLT ;DS!AS
021370 4$: TST @RSWC ;TEST WC
021374 BEQ +4 ;WORD COUNT DID OVERFLOW
021376 HLT ;WC ;SHOULD = 0
021400 MOV UNNUM, GOOD ;GET CORRECT
021404 BIS #100, GOOD ;ANS OF CS2
021410 MOV @RSCS2, BAD ;GET CS2
021414 CMP GOOD, BAD ;IS CS2 CORRECT?
021416 BEQ +4 ;YES
021420 HLT ;GOOD = CORRECT ANS FOR CS2
021422 MOV @RSBA, BAD ;FETCH CURRENT ADDRESS
021426 MOV @#OBUFSV, GOOD ;WHAT RSBA SHOULD EQUAL
021432 ADD #2000, GOOD ;UPDATE IT
021436 CMP BAD, GOOD ;IS RSBA CORRECT
021440 BEQ +4 ;YES EXECUTE CONTINUE
021442 HLT ;RSBA FAILED TO INCREMENT
021444 CMP #4, RS04DT ;RS03LA?
021452 BNE 2$ ;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$: CMP #64, @RSDA ;WAS RSDA INCREMENTED
021520 BEQ +4 ;RSDA OK
021522 HLT ;RSDA SHOULD CONTAIN A 64
021524 3$: 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 TTVEC: CLR R2 ;CLEAR UNIT #
021570 CLC
021572 MOV #401,WORK
021600 1S: BIT WORK,ARSAS ;DID THIS DRIVE INT?
021606 BNE 2S ;YES
021610 INC R2 ;UPDATE UNIT #
021612 CLC
021614 ROL WORK
021620 BCC 1S
021622 HLT AS ;WHY DID WE INT WITH NO ATA???
021624 2S: MOV R2,ARSCS2 ;GET DRIVE
021630 CMP #110600,ARSDS ;DID PIP CLEAR?
021636 BEQ +4 ;YES
021640 HLT DS:AS ;PIP BIT DID NOT CLEAR
021642 CMP #104230,ARSCS1 ;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,ARSAS ;CLEAR AS
021670 MOV #100,ARSCS1 ;SET IE
021676 MOV #WTDV2,(SP) ;RETURN TO WAIT
021702 RTI
021704 WTDV1: MOV #340,ARPS
021712 MOV #500,SP ;CLEAR STACK
021716 MOV RSVCP,ARSVEC ;RESTORE INT VECTOR
021724 CLR ARSVCP
021730 MODDON: SCOPE ;DONE
021732 MOV TIMSV,TIMES ;RESTORE LOOP COUNT
021740 MOV #340,ARPS ;RESTORE PS
021746 MOV #1,ICNT ;FUZE TEST NUMBERS
021754 OUT: JMP ARTRYNX ;TEST NEXT DRIVE
.SBTTL SDONE - BELL AND SCOPE ROUTINE

021760 DONE: SCOPE ;TERMINATIONG SCOPE FOR LOOPING
021762 ADD #1,PCNT+2 ;ADD 1 TO THE PASS COUNT
021770 ADC PCNT ;MAKE IT DOUBLE PREC.
021774 BIT #SW10,ASWR ;RING THE BELL?
022002 BNE 4S ;NO!
022004 TYPE .+2 ;ASCIZ <BELL><177>
022014 4S: MOV AR42,RO ;GET MONITOR ADDRESS
022020 BEQ SEND1 ;IF NONE
022022 RESET
022024 SENDAD: JSR 7,(0) ;GO TO MONITOR
022026 SEND1: JMP 240,240 ;SAVE ROOM FOR ACT11
022034 ;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 INLINE 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 TYPEN?
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,2TPB	:TYPE A FILLER
022130		TSTB	2TPS	:DONE YET?
022134		BPL	-4	:NO - WAIT
022136		DEC	R4	:DEC COUNT
022140		BNE	5\$	:LOOP UNTIL 0
022142	4\$:	MOVB	(R5)+,2TPB	:LOAD AND TYPE THE CHARACTER
022146		TSTB	2TPS	: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 SSCOPE - 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:"

```

022220 .SCOPE: KBOIN          :GO CHECK FOR ?G
022222 BIT @SW8,@SWR      :LOOP ON SPEC. TEST?
022224 BEQ @S          :NO LOOP ON SPEC. TEST
022226 CMPB @SWR,ICNT     :ON RIGHT TEST? @SW7-0*
022228 BEQ .OVER        :NOT RIGHT TEST
022230 1S: BIT @SW14,@SWR :LOOP ON TEST?
022232 BNE .KIT         :LOOP ON TEST IS SET
022234 BR @S           :SKIP - NOP FOR XOR TESTER
022236 MOV @84,-(6)      :PUSH @84 ON STACK
022238 MOV @48,@84      :SET FOR TIMEOUT
022240 TST @8177060      :ERROR ON XOR?
022242 MOV (6)+,@84     :POP STACK INTO @84
022244 BR .SVLAD       :NO ERROR - GO TO NEXT TEST
022246 4S: CMP (6)+,(6)+ :CLEAR STACK
022248 MOV (6)+,@84     :POP STACK INTO @84
022250 BR .KIT         :ERROR - LOOP ON TEST
022252 3S: BIT @SW11,@SWR :KILL ITERATIONS
022254 BNE .SVLAD       :YES - KILL ITERATIONS
022256 TSTB ICNT+1      :FIRST ONE?
022258 BEQ @S         :BRANCH IF FIRST
022260 CMPB TIMES,ICNT+1 :DONE?
022262 BGT .KIT         :BRANCH IF NOT
022264 2S: MOVB @1,ICNT+1 :FIRST ITERATION
022266 .SVLAD: INCB ICNT :COUNT TEST NUMBERS
022268 MOV (6),LAD      :SAVE LOOP ADDRESS
022270 MOV ICNT,@DISPLAY  :DISPLAY TEST NO. AND ITERATION COUNT
022272 RTI            :RETURN

022364 .KIT: INCB ICNT+1 :INC THE ITERATION COUNT
022370 .OVER: MOV ICNT,@DISPLAY :SET UP DISPLAY
022376 TST LAD           :FIRST ONE?
022402 BEQ .SVLAD       :YES
022404 MOV LAD,(6)      :FUDGE RETURN ADDRESS
022410 RTI            :FIXES PS

022412 TIMES: 100    :RUN 100 TIMES
    
```

.SBTTL SHLT - HLT ROUTINE (ERROR TYPEOUT)

: THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE  
: ADDRESS OF THE "HLT". IT ALSO COUNTS THE NUMBER OF ERRORS  
: AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,  
: "HALT" ON ERROR, AND INHIBIT TYPEOUTS. AN OPTIONAL ARGUMENT  
: (HLT+3) WILL BE PLACED IN ".HLTCT:" FOR ADDITIONAL TYPEOUTS.

```

022514 .HLT: KBDIN          :GO CHECK FOR IG
022516 BIT            #SW10,2SWR :BELL ON ERROR?
022518 BEQ          15      :NO - SKIP
022520 TYPE        BELL     :RING BELL
022522 15: INC        ERRORS  :COUNT THE NUMBER OF ERRORS
022524 BIT            #SW13,2SWR :SKIP TYPEOUT IF SET
022526 BNE        25      :SKIP TYPEOUTS
022528 TYPE        +2      :ASCIZ '<15><12>'
022530 MOV        (6),HLTADR :PUT ADDRESS OF INSTRUCTION ON STACK
022532 SUB        #2,HLTADR  :FUDGE ADDRESS
022534 MOVB       SHLTADR,.HLTCT :GET HLT ARGUMENT
022536 MOV        HLTADR,-(6) :PUT HLTADR ON STACK
022538 TYPE0      :TYPE STACK IN OCTAL
022540 TYPE        +2      :ASCIZ " "
022542 JSR        PC,RSREG   :GO TO USER ERROR ROUTINE
022544 25: TST        2SWR   :HALT ON ERROR
022546 BPL          +4      :SKIP IF CONTINUE
022548 HALT       :HALT ON ERROR!
022550 BIT            #SW9,2SWR :CHECK FOR INHIBIT LOOP ON ERROR
022552 BNE        35      :SKIP IF LOOP ON ERROR
022554 CLRB       ICNT+1    :CLEAR ITERATION COUNT
022556 RTI          :RETURN
022558 35: JMP        .KIT   :LOOP ON TEST UNTIL NO ERRORS

022552 .HLTCT: 0          :HLT ARGUMENT

```

.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 TYPOCT, TYPBIT, OR TYPOCS MACRO'S.

```

022554 .TYPEB: MOV      #170101,.PR      ; SET BIT FLAG AND 16 CHARACTER COUNT
022556 BR          .PTIT              ; NOW TYPE IT IN BIT FORM
022564 .TYPEO: MOVB     #1,.PR          ; SET ZERO FILL SWITCH
022572 BR          .+6                ; SKIP
022574 .TYPES: CLR     .PR            ; SUPPRESS LEADING ZERO'S
022600 MOVB     #-6,.PR+1            ; SET COUNT
022606 .PTIT:
022608 MOV      R4,-(6)                ; PUSH R4 ON STACK
022610 MOV      R5,-(6)                ; PUSH R5 ON STACK
022612 MOV      10(6),R5             ; GET THE DATA
022616 MOV      #.PR+2,R4           ; SET POINTER TO FIRST ASCII CHAR.
022622 CLRB    (4)                  ; CLEAR FIRST BYTE
022624 BR          .PRF              ; ROTATE FIRST BIT
022626 .PRL: CLRB    (4)              ; CLEAR BYTE OF CHARACTER
022630 BIT      #100,.PR             ; BIT TYPING MODE?
022636 BNE     .PRF                  ; YES - SKIP 2 ROTATES
022640 ROL      R5                    ; ROTATE BIT INTO C
022642 ROLB   (4)                  ; PACK IT
022644 ROL      R5                    ; ROTATE BIT INTO C
022646 ROLB   (4)                  ; PACK IT
022650 .PRF: ROL      R5              ; ROTATE BIT INTO C
022652 ROLB   (4)                  ; PACK IT
022654 TSTB   (4)                  ; IS IT ZERO?
022656 BEQ    .+6                  ; SKIP INC
022660 INCB   .PR                  ; SET FILL SWITCH
022664 TSTB   .PR                  ; CHECK FILL SWITCH
022670 BEQ    .+6                  ; SKIP BITSET
022672 BISH   #'0,(4)+              ; MAKE INTO ASCII CHAR
022676 INCB   .PR+1                ; INC COUNT
022702 .PRL: REPEAT
022704 CMP     #.PR+2,R4            ; EMPTY BUFFER?
022710 BNE     .+6                  ; SKIP IF NOT
022712 MOVB   #'0,(4)+              ; LOAD 1 ZERO
022716 CLRB   (4)                  ; NULL TERMINATOR
022720 TYPE   .PR+2                ; TYPE IT
022724 MOV     (6)+,R5               ; POP STACK INTO R5
022726 MOV     (6)+,R4               ; POP STACK INTO R4
022730 MOV     2(6),4(6)            ; GET RID OF
022736 MOV     (6)+,(6)             ; DATA WORD
022740 RTI
022742 .PR: .BLKW 12                ; COUNT, SWITCH, AND OUTPUT BUFFER
    
```

.SBTTL SPOWER - 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 1$: INC      R1 : WAIT FOR THE INC
023042      BNE     1$ : 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   ..+2 : .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                    SRDOCT - 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.

```

023124 .RDOCT: MOV      (6), -(6)      : MOVE THE PC
023126 MOV      4(6), 2(6)      : MOVE THE PS
023128 MOV      R1, -(6)       : PUSH R1 ON STACK
023130 MOV      R2, -(6)       : PUSH R2 ON STACK
023132 MOV      R3, -(6)       : PUSH R3 ON STACK
023134 4$: RDLIN              : READ A LINE INTO INPUT
023136 CLR      R1            : INIT DATA WORD
023138 CLR      CTN           : CLEAR COUNT WORD
023140 MOV      @INPUT, R3     : INIT POINTER
023142 1$: MOV      (3)+, R2    : GET A BYTE
023144 CMP      @15, R2        : WAS IT A CR?
023146 BEQ      2$           : GET OUT IF YES
023148 CMP      @0, R2        : CHECK FOR 0 OR GREATER
023150 BGT      3$           : ERROR - LESS THAN 0
023152 CMP      @7, R2        : CHECK FOR 7 OR LESS
023154 BLT      3$           : ERROR - GREATER THAN 7
023156 ROR      R2           : GET
023158 ROR      R2           : INTO
023160 ROR      R2           : POSITION
023162 ROL      R1           : FIRST BIT
023164 ROL      R2           : GET
023166 ROL      R1           : SECOND BIT
023168 ROL      R2           : GET
023170 ROL      R1           : THIRD BIT
023172 INC      CTN          : YES HE TYPED SOMETHING
023174 BR      1$           : LOOP
023176 2$: MOV      R1, 12(6)  : SAVE THE RESULT
023178 MOV      (6)+, R3       : POP STACK INTO R3
023180 MOV      (6)+, R2       : POP STACK INTO R2
023182 MOV      (6)+, R1       : POP STACK INTO R1
023184 RTI                  : RETURN

023244 3$: TYPE      4$+2       : .ASCIZ "???"(15)(12)
023246 BR      4$           : TRY AGAIN
023254
  
```

.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 TSTB @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$: TYPE .+2 ;.ASCIZ "?"(15)(12)
023350 BR 1$ ;ZAP THE BUFFER AND LOOP
023352 3$: MOVB (5), #0 ;SET UP FOR TYPING
023356 TYPE 3$+2 ;ECHO IT
023362 CMPB @15, (5)+ ;CHECK FOR RETURN
023366 BNE 2$ ;LOOP IF NOT RETURN
023370 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 .TRP: 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	.CLROK	:CLROK	=	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,@RSMC      :MOV SWR INTO WORD COUNT REG
023500      MOV @RSMC,@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,@RSBA      :MOV SWR INTO CURRENT ADDR REG
023524      MOV @RSBA,@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,@RSDA      :MOV SWR INTO DISK ADDR REG
023550      MOV @RSDA,@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,@RSDS      :MOV SWR INTO DRIVE STATUS
023574      MOV @RSDS,@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,@RSER      :LOAD ER REG
023620      MOV @RSER,@DISPLAY  :DISPLAY IT IN LIGHTS
023626      BR IS

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

```

:RSCS2 REGISTER
023640 SRCS2: MOV #500,SP :INIT THE STACK
023644 1S: KBDIN :CTW
023646 MOV @SWR,@RSCS2 :LOAD CS2
023654 MOV @RSCS2,@DISPLAY :DISPLAY IT
023662 BR 1S

:RSAS REGISTER
023664 SRAS: MOV #500,SP :INIT THE STACK
023670 1S: KBDIN :CTW
023672 MOV @SWR,@RSAS :LOAD RSAS
023700 MOV @RSAS,@DISPLAY :DISPLAY IT
023706 BR 1S

:RSMR REGISTER
023710 RSMR: MOV #500,SP :INIT THE STACK
023714 1S: KBDIN :CTW
023716 MOV @SWR,@RSMR :LOAD RSMR
023724 MOV @RSMR,@DISPLAY :DISPLAY IT
023732 BR 1S

:DISK CONTROL STATUS REGISTER
023734 SRSCS1: MOV #500,SP :INIT THE STACK
023740 1S: KBDIN :CTW
023742 MOV #340,@PS :LOCK UP INTERRUPTS
023750 MOV #177777,@RSWC :SET WORD COUNT -1 WORD
023756 MOV @#OBUFSV,@RSBA :SET UP CURRENT ADDRESS
023764 MOV @SWR,@RSCS1 :MOV SWR INTO CONTROL REG
023772 BIT @BIT0,@RSCS1 :IS FUNCTION BITS SET
024000 BEQ 1S :FUNCTION BITS NOT SET
024002 2S: TSTB @RSCS1 :TEST FOR DISK READY
024006 BPL 2S :DISK STILL NOT READY
024010 BR 1S :DISK NOT BUSY SECT NEW CR

```

```

: 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 TKSEL: MOV #500,SP ;SET STACK
024016 NOP
024020 TYPE ..+2 ;.ASCIZ <15><12>"TYPE UNIT # "
024044 RDOCT
024046 MOV (6)+,UNNUM
024052 TYPE ..+2 ;.ASCIZ <15><12>"TYPE POSITIVE (OCTAL) WC "
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 ;INIT SWITCHLESS
024176 TK1: MOV @SWR,WORK2 ;SAVE SWR
024204 TK2: CLRDK ;CLEAR ALL RS REG
024206 BIS #BIT3,@RSCS2 ;SET BAI
024214 MOV @SWR,WORK1 ;GET SWR FOR DSK ADDR
024222 BIC #17000,WORK1 ;CLEAR UNIT #
024230 TKKS: MOV WORK1,@RSDA ;LOAD THE DA
024236 MOV WORK,@RSCW ;LOAD WORD COUNT
024244 BIT #BIT12,@SWR ;READ?
024252 BEQ WTE ;NO
024254 MOV #OUTBUF+2,@RSBA ;LOAD CURRENT ADDRESS
024262 MOV #71,@RSCS1 ;GO AND READ
024270 WT: TSTB @RSCS1 ;TEST FOR READY
024274 BPL -4
024276 BR SWRCHG
024300 WTE: MOV #OUTBUF,@RSBA
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 ;CHECK FOR NEW VALUE
024336 CMP @SWR,WORK2 ;DID SWR CHANGE?
024344 BNE TK1 ;YES
024346 BR TK2 ;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      ;.+2      ;.ASCIZ <15><12>"LOAD SW WITH UNIT # AND CONT"
#SWREG,SWR
7$        7$        ;CHECK IF SWITCHLESS CPU
HALT
BR        8$        ;GO AROUND TRAP CALL
CNTLU
8$:      MOV        @SWR,UNNUM ;GET SWREG VALUE
          CLRDK     ;GET UNIT #
          CLR        ;CLEAR ALL REG
          MOV        @OUTBUF,@RSBA ;PUT A 0 INTO DATA BUFFER
          MOV        @7700,@RSDA ;SETUP REG TO
          MOV        @-1,@RSMC    ;TO A WRITE
          MOV        @61,@RSCS1
6$:      TSTB      @RSCS1      ;WAIT FOR DONE
          BPL        6$
          TYPE      ;.+2      ;.ASCIZ <15><12>"SET WRITE LOCK ENABLE AND CONTINUE"
HALT
1$:      MOV        @0,@RSDA
          CMP        @14600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=14600
          MOV        @100,@RSDA
          CMP        @10600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=10600
          TYPE      ;.+2      ;.ASCIZ <15><12>"SET WRT LOC SW 0 AND CONT"
          HALT
2$:      CMP        @14600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=14600
          MOV        @300,@RSDA
          CMP        @10600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=10600
          TYPE      ;.+2      ;.ASCIZ <15><12>"SET WRT LOC SW 1 AND CONT"
          HALT
3$:      CMP        @14600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=14600
          MOV        @700,@RSDA
          CMP        @10600,@RSDS
          BEQ        +4
          HLT        !DS!DA      ;DS SHOULD=10600
          TYPE      ;.+2      ;.ASCIZ <15><12>"SET WRT LCK SW 2 AND CONT"
          HALT
```

```

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, @RSWC
025422      MOV      #61, @RSCS1      ;TRY TO WRITE
025430 7S:  TSTB    @RSCS1      ;WAIT FOR DONE
025434      BPL      7S
025436 5S:  TSTB    @RSCS1      ;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, @RSCW
025532      MOV      #71, @RSCS1 ; DO A READ TO SEE IF DISK DID
025540      BS:    TSTB   @RSCS1 ; ACTUALLY GET WRITTEN ON TO
025544      BPL      BS          ; WAIT FOR DONE
025546      TST      OUTBUF      ; IS DATA STILL 0 ON THE DSK?
025552      BEQ      +4          ; YES
025554      HLT      !DS          ; NO DSK DID GET WRITTEN ONTO WITH WLE SET
025556      TYPE    ,.+2        ; .ASCIZ <15><12>"RESET ALL W/L SWITCHES"
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	BS	:	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	BS:	TYPE	.+2	:	.ASCIZ " CS1="
026146		MOV	ARSCS1,-(6)	:	PUT ARSCS1 ON STACK
026152		TYPE0		:	TYPE STACK IN OCTAL
026154	1S:	TYPE	.+2	:	.ASCIZ " ER="
026166		MOV	ARSER,-(6)	:	PUT ARSER ON STACK
026172		TYPE0		:	TYPE STACK IN OCTAL
026174	2S:	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	3S	:	NO
026234		TYPE	.+2	:	.ASCIZ " AS="
026246		MOV	ARSAS,-(6)	:	PUT ARSAS ON STACK
026252		TYPE0		:	TYPE STACK IN OCTAL
026254	3S:	BIT	#BA,.HLTCT	:	TYPTXT BUS ADDRESS
026262		BEQ	4S	:	NO
026264		TYPE	.+2	:	.ASCIZ " BA="
026276		MOV	ARSBA,-(6)	:	PUT ARSBA ON STACK
026302		TYPE0		:	TYPE STACK IN OCTAL
026304	4S:	BIT	#DA,.HLTCT	:	TYPTXT DA ?
026312		BEQ	5S	:	NO
026314		TYPE	.+2	:	.ASCIZ " DA="
026326		MOV	ARSDA,-(6)	:	PUT ARSDA ON STACK
026332		TYPE0		:	TYPE STACK IN OCTAL
026334	5S:	BIT	#WC,.HLTCT	:	TYPTXT WC?
026342		BEQ	6S	:	NO
026344		TYPE	.+2	:	.ASCIZ " WC="
026356		MOV	ARSWC,-(6)	:	PUT ARSWC ON STACK
026362		TYPE0		:	TYPE STACK IN OCTAL

```

026364 6S: BIT 8DS .MLTCT :DRIVE STATUS
026372 BEQ PTDONE :NO
026374 TYPE A+2 :ASCIZ "DS="
026406 MOV 8RSDS, -(6) :PUT 8RSDS ON STACK
026412 TYPE0 :TYPE STACK IN OCTAL
026414 JMP PTDONE :GET OUT
026416 SEEC: BIC 8200 .MLTCT :CLEAR COMMON BIT
026418 BIT 8DT .MLTCT :TYPTXT DRIVE TYPE?
026420 BEQ 9S :NO
026422 TYPE A+2 :ASCIZ "DT="
026424 MOV 8RSDT, -(6) :PUT 8RSDT ON STACK
026426 TYPE0 :TYPE STACK IN OCTAL
026428 9S: BIT 8DJ .MLTCT :TYPTXT DATA BUFFER
026430 BEQ 10S :NO
026432 TYPE A+2 :ASCIZ "DB="
026434 MOV 8RSD8, -(6) :PUT 8RSD8 ON STACK
026436 TYPE0 :TYPE STACK IN OCTAL
026438 10S: BIT 8MR .MLTCT :TYPTXT MN?
026440 BEQ 11S :NO
026442 TYPE A+2 :ASCIZ "MR="
026444 MOV 8RSMR, -(6) :PUT 8RSMR ON STACK
026446 TYPE0 :TYPE STACK IN OCTAL
026448 11S: BIT 8LA .MLTCT :TYPTXT LA?
026450 BEQ PTDONE :NO
026452 TYPE A+2 :ASCIZ "LA="
026454 MOV 8RSLA, -(6) :PUT 8RSLA ON STACK
026456 TYPE0 :TYPE STACK IN OCTAL
026458 PTDONE: BIS 8BIT15, ONCEE :SET FORND ERROR FLAG
026460 RTS PC

026576 WAITRY: CLR WORK :CLEAR COUNTER
026578 1S: TSTB 8RSCS1 :TEST READY
026580 BMI 2S :OK CONT
026582 INC WORK :UPDATE COUNTER
026584 TST WORK :DONE YET?
026586 BEQ 3S :READY DID NOT COME UP
026588 BR 1S :CONTINUE WAITING
026590 2S: ADD 82, (SP) :UPDATE RETURN PC
026592 3S: RTS PC :RETURN

```

:RANDOM DATA GENERATOR SUBROUTINE

```

026632 RANDOM: MOV LONUM,LOSAY
026634 MOV MINUM,HISAV
026640 RAND1: MOV LONUM,R0
026642 MOV MINUM,R4
026644 MOV #7,R3
026646 CLR R2
026648 SHIFT: ASL R0
026650 ROL R4
026652 ROL R2
026654 DEC R3
026656 BNE SHIFT
026658 ADD LONUM,R0
026660 ADC R4
026662 ADD MINUM,R4
026664 ROL R2
026666 ADD #1057,R0
026668 ROL R4
026670 ADC R2
026672 ADD #47401,R4
026674 ADC R2
026676 ADD #6,R2
026678 ADD #2,R0
026680 ADC R4
026682 MOV R0,LONUM
026684 MOV R0,(R1)+
026686 DEC WORK
026688 BEQ EXGEN
026690 MOV R4,MINUM
026692 MOV R4,(1)+
026694 DEC WORK
026696 BNE RAND1
026698 EXGEN: RTS
026700 LONUM: 0
026702 HINUM: 0
027000 LOSAV: 0
027002 HISAV: 0
027004 RANEND:

027004 .SUSWR: BIT #BIT0,SWI
027006 BNE XXX
027008 MOV 6,-(SP)
027010 MOV 4,-(SP)
027012 MOV #15,4
027014 CMP #-1,2SWR
027016 BEQ 25
027018 BR 35
027020 15: CMP (SP)+,(SP)+
027022 25: MOV #SWREG,SWR
027024 MOV #DISPREG,DISPLAY
027026 35: CMP #SWREG,SWR
027028 BNE 45
027030 TST 42
027032 BNE 45
027100 CNTLU
    
```

```

:SET UP R0 WITH 5 DIGITS LOW
:SET UP R1 WITH 5 DIGITS HIGH
:SET UP SHIFT COUNT
:CLEAR R2
:SHIFT R0 LEFT AND
:ROTATE CARRY INTO LSB OF R1 INTO
:ROTATE CARRY OUT OF R1 INTO R2
:DECREMENT R3
:CONTINUE SHIFT LOOP
:ADDN IN NUMBER TO MAKE X 129
:PROPOGATE CARRY
:ADDN IN NUMBER TO MAKE X 129
:PROPOGATE CARRY
:ADDN LOW CONSTANT
:PROPOGATE CARRIES
:PROPOGATE AGAIN
:ADDN HIGH CONSTANT
:PROPOGATE CARRY
:ADDN 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
    
```

```

027102 4S:   MOV      (SP)+,4      :REPLACE 4 FROM STACK
027106 4S:   MOV      (SP)+,6      :REPLACE 6 FROM STACK
027112 XXX:   BIS      @BIT0,SWI   :SET THE BEENHEREBIT
027120 RTI
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?
027140 BNE OKT      :NO GET OUT
027142 TSTB     @TKS          :HAVE A CHARACTER
027146 BPL OKT      :NO GET OUT
027150 MOV      @TKB,.MSG      :
027156 BIC      @177600,.MSG   :STRIP OFF GARBAGE
027164 CMPB     @7,.MSG        :DO WE HAVE A 'G'
027172 BNE OKT      :NO GET OUT
027174 TYPE     ..+2        :..ASCIZ <15><12>"IG"
027206 .CNTLU: TYPE     ..+2        :..ASCIZ <15><12>"SWR= "
027206 MOV      @SWREG,-(6)   :PUT SWREG ON STACK
027222 TYPE0    ..+2        :TYPE STACK IN OCTAL
027226 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: 0
027300      0
027302      0
027304      0
027306      0
027310      0
027312      0
027314      0
027316      0
027320      0
027322      0
027324      0
027326      0
027330      0
027332      0
027334      0
027336      0
027340 ILFTB2: 0
027342      0
027344      0
027346      0
027350      0
027352      0
027354      0
027356      0
027360      0
027362 OUTBUF: .BLKW 300
030162 INBUF:  .BLKW 300
030762 PARITY: 0      :USED FOR PARITY TEST 67
          .END

```

## SYMBOL TABLE

ADDCF	012120	DOMINT	020756	KBDIN	= 104416	PCNT	001004	SECT	017330
AS	= 000100	DRCLR	005754	KIPAR0	= 172340	PGE	= 002000	SEEC	026420
ATA	= 100000	DRY	= 000200	KIPAR1	= 172342	PGETST	013414	SETPAT	016502
ROB1	001166	DS	= 000040	KIPAR2	= 172344	PGTRAP	020520	SHIFT	026664
BA	= 000020	DT	= 000240	KIPAR3	= 172356	PI?	= 020000	SILO	004632
BAD	=x000000	DVNUM	001346	KIPAR4	= 172300	PATP	012422	SILOB	004422
BAI	= 000010	DVTAB	022042	KIPAR5	= 172302	PATP1	012424	SILOFL	005004
BAITST	007754	ER	= 000002	KIPAR6	= 172304	PS	= 177776	SRAS	023664
BEGIN	001202	ERINT	020760	KIPAR7	= 172316	PSW	= 177776	SRC52	023640
BELL	= 000007	ERR	= 040000	LA	= 000204	PTDONE	026566	SRSBA	023510
BITBA	004350	ERRINT	021050	LA0	001010	QES	020440	SRSOS1	023734
BITCS2	004234	ERRORS	001002	LASTSC	017164	QESDON	020534	SRSOA	023534
BITST	004162	ERRR	012002	LATDON	012046	QO	= 000001	SRSOS	023560
BITWC	004300	ERTST	016034	LATST	011724	RANDOM	026632	SRSER	023604
BITO	= 000001	EXGEN	026772	LBT	= 002000	RAND1	026646	SRSLA	023630
BIT1	= 000002	EXREF	025662	LONUM	026774	RANEND	027004	SRSWC	023464
BIT10	= 002000	EXTRP	020410	LOP1	003772	RANNU	001150	SRO	= 177572
BIT11	= 004000	EXTTRP	020416	LOP2	004042	RANTS	003742	SRAT	000230
BIT12	= 010000	EXTTST	017674	LOP3	004112	RDLIN	= 104412	STTEST	001432
BIT13	= 020000	EXTT1	020216	LOSAY	027000	RDOCT	= 104410	SUSWR	= 104420
BIT14	= 040000	EXT1	020420	MAXREF	025714	ROST	006376	SWI	027122
BIT15	= 100000	FILCHR	001014	MAXRF1	025716	ROSTB	007154	SWR	001026
BIT2	= 000004	FINDM	025630	MEMOUT	020424	RMRT1	013614	SWRCHG	024316
BIT3	= 000010	FINTST	021064	MODDON	021730	RMRT2	013772	SWREG	000176
BIT4	= 000020	FLOTBA	003052	MODNUM	021066	RMRT3	014102	SW10	= 002000
BIT5	= 000040	FLOTDA	003352	MPRO	= 172100	RMRT4	014212	SW11	= 004000
BIT6	= 000100	FLOTWC	003212	MR	= 000220	RMAS	001122	SW12	= 010000
BIT7	= 000200	FUNDO	016616	MULTI	= 001316	RSBA	001112	SW13	= 020000
BIT8	= 000400	GOOD	=x000001	N	= 000123	RSBAB	001146	SW14	= 040000
BIT9	= 001000	HALT	025146	NED	= 010000	RSCS1	001104	SW15	= 100000
BLOCK	010206	HINUM	026776	NEDDON	015420	RSCS1B	001140	SW8	= 000400
BPORTT	001170	HISAV	027002	NEDTST	015242	RSCS2	001106	SW9	= 001000
CHKDV	001650	HLT	= 104000	NND	015502	RSCS2B	001142	TIMES	022412
CLROK	= 104414	HLTADR	001012	NNOOP	005244	RSDA	001114	TIMSV	001164
CNTLU	= 104422	IAERR	015126	NNOP21	005562	RSD8	001126	TKB	001022
CSTST	012050	ICNT	001000	NOOP	005120	RSDS	001116	TKKS	024230
CS1	= 000001	IE	= 000100	NOOP21	005424	RSDT	001132	TKS	001020
CS2	= 000200	ILFDN	010762	NOPAR	012442	RSEI	001120	TKSEL	024012
CTN	027274	ILFDNE	011332	NOWGO	002022	RSLA	001124	TK1	024176
DA	= 000004	ILFTB2	027340	NXM	007526	RSMR	001130	TK2	024204
DA0	= 001000	ILF67	011334	NXMTSM	010060	RSMRR	023710	TMEOUT	012234
DAOTST	015512	ILLFUN	010764	OBUSV	001102	RSREG	026064	TPB	001024
DAOTT	015670	ILLTAB	027276	OKT	027270	RSVCPS	001136	TPS	001016
DB	= 000210	ILLS1	010506	ONCEE	001160	RSVEC	001134	TRE	= 040000
DCK	= 100000	INBUF	030162	OOUT	010504	RSWC	001110	TRY	001326
DCKTST	014362	INCW	014574	OR	= 000200	RSWCB	001144	TRYNX	001634
DISPLA	001030	INPUT	023400	OUT	021754	RSWCT	006626	TSTAGN	012266
DISPRE	000174	INTDON	020640	OUTBUF	027362	RSO4DT	001162	TSTNEM	010120
DKADR	015052	INTR4	020642	PARITY	030762	RW	= 000006	TSTVEC	021336
DLT	= 100000	INTR5	020536	PART	012240	SAVBAD	001100	TST1	001314
DONCS	012236	INT112	020632	PARTST	011646	SAVEE	001172	TST10	003050
DONE	021760	INT114	020744	PATST	016330	SC	= 100000	TST100	015050
DONEE	002016	IR	= 000100	PATTST	016770	SCOPE	= 104400	TST101	015124

TST102	015240	TST25	003616	TST6	002626	WC	= 000010	.CNTLU	027206
TST103	015510	TST26	003644	TST60	010504	WCE	= 040000	.HLT	022414
TST104	015666	TST27	003702	TST61	010764	WCEDNB	= 013354	.HLTCT	022552
TST105	016032	TST3	002274	TST62	011332	WCEDNE	= 012770	.ILLUP	023112
TST106	016326	TST30	003740	TST63	011644	WCEDON	= 013012	.KBDIN	027124
TST107	016500	TST31	004160	TST64	011722	WCEDOS	= 013376	.KIT	022364
TST11	003102	TST32	004232	TST65	011736	WCFTB	= 013052	.MSG	027272
TST110	016614	TST33	004276	TST66	012046	WCFTSB	= 013022	.OVER	022370
TST111	016766	TST34	004346	TST67	012036	WCFTST	= 012460	.POWER	022766
TST112	017162	TST35	004420	TST7	002766	WCETT	= 012466	.POWUP	023032
TST113	017326	TST36	004522	TST70	012456	WORK	= 001174	.PR	022742
TST114	017672	TST37	004630	TST71	013020	WORK1	= 001176	.PRF	022650
TST115	020436	TST4	002506	TST72	013412	WORK2	= 001200	.PRL	022626
TST116	020534	TST40	005002	TST73	013612	WRCKT	= 006564	.PTIT	022606
TST117	020640	TST41	005116	TST74	013770	WRCKTB	= 007350	.PUVEC	023120
TST12	003126	TST42	005242	TST75	014100	WRTLCK	= 024352	.QQ	= 000024
TST120	020756	TST43	005422	TST76	014210	WRTST	= 006150	.ROLIN	023256
TST121	021064	TST44	005560	TST77	014360	WRTSTB	= 006736	.RDOCT	023124
TST122	024350	TST45	005752	TTAGG	002276	WT	= 024270	.SAVR6	023116
TST13	003210	TST46	006146	TTMOUT	010502	WTDV	= 021334	.SCOPE	022220
TST14	003242	TST47	006374	TTVEC	021566	WTDV1	= 021704	.SUSWR	027004
TST15	003266	TST5	002600	TYPE	= 104402	WTDV2	= 021564	.SVLAD	022344
TST16	003350	TST50	006562	TYPE0	= 104404	WTE	= 024300	.SWOPT	= 167400
TST17	003402	TST51	006734	TYPES	= 104406	WMP	= 000004	.TBIT	022040
TST2	002052	TST52	007152	UNCMP	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	.CLRDK	= 030764	.TYPEO	022564
TST24	003560	TST57	010204	WAITRY	026576			.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