

IDENTIFICATION

0 1

SEQ 0001

MODEL: AC-80150-MK  
TITLE: PERSBIO RM70-RS03-RS04 DATA RELIABILITY  
DIAGNOSTIC  
DATE PREPARED: AUG 1978  
MAINTAINER: DIAGNOSTIC GROUP  
AUTHORS: STANLEY HARACKIEWICZ

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 manual.

The software described in this document is furnished to the purchaser under a license for use on a single computer system and can be copied (with inclusion of DIGITAL's copyright notice) only for use in such system, except as may otherwise be provided in writing by DIGITAL.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

Copyright (C) 1975, 1978 by Digital Equipment Corporation

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
  - 2.1 EQUIPMENT
  - 2.2 PRELIMINARY PROGRAMS
- 3. LOADING PROCEDURE
- 4. STARTING PROCEDURE
  - 4.1 CONTROL SWITCH SETTINGS
  - 4.2 STARTING ADDRESS
- 5. OPERATIONAL SWITCH SETTINGS
  - 5.1 DATA RELIABILITY TEST MODE
  - 5.2 CONVERSATION MODE
  - 5.3 ROUTINE ABSTRACTS
  - 5.4 SUBROUTINE ABSTRACTS
- 6. ERRORS
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
  - 8.1 EXECUTION TIME
  - 8.2 STACK POINTER
  - 8.3 POWER FAIL

## 1. ABSTRACT

This Diagnostic was designed to test RS03 and RS04 drives.

The CERSB Disk Data Test is a series of address and data reliability routines which verify to the user that the controller (RH70) and the disks (RS03 or RS04) are operating correctly. This test should be used in conjunction with the DERSA diagnostic.

### NOTE

This program will destroy all data on the disks. Turn off all drives that you do not want to test.

## 2. REQUIREMENTS

### 2.1 Equipment

PDP11 standard computer with a minimum of 8K of memory, and an RH70 controller with an RS03 or an RS04 disk.

### 2.2 Preliminary Programs

DZRSA

## 3. LOADING PROCEDURE

Use standard procedure for ABS tapes.

#### 4. STARTING PROCEDURE

##### 4.1 Control Switch Settings

See 5.1 (all down for worst case testing)

##### 4.2 Starting Address

Program and/or operator action

Load program into memory using ABS loader.

##### 1. Starting address 200.

- A. Set switches (see sec 5.1). All down for worst case
- B. 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.
- C. Press start.

The program will now map the data buffers in 4K segments on -A- port for all memory. It will then type out the parameters of the data buffers. The program will only do this the first time it is started, for it stores these addresses and continues using them. To have the program remap the system, the program must be reloaded. All of memory will be tested. You may enter conversation mode and put data buffer where you wish and what ever size you wish.

### 5. OPERATIONAL SWITCH SETTINGS

Switch Settings Are:

SW<15> = 1 ..... HALT ON ERROR  
SW<14> = 1 ..... LOOP ON FUNCTION  
SW<13> = 1 ..... INHIBIT PRINTOUT  
SW<12> = 1 ..... INHIBIT COMPARISON  
WITH THIS SWITCH SET, THE  
PROGRAM WILL NOT COMPARE THE  
DATA IT READ FROM THE DISK WITH  
THE KNOWN GOOD DATA.  
SW<11> = 1 ..... HALT ON COMPLETION OF TRANSFER  
SW<10> = 1 ..... ENTER CONVERSATION MODE  
SW<09> = 1 ..... LOOP ON ERROR  
SW<07> = 1 ..... WAIT IN WAIT MODE  
PROGRAM RUNS IN A BACKGROUND TEST  
WHILE WAITING FOR INTERRUPT, WITH  
SW SET PROGRAM WAITS IN A WAIT  
INSTRUCTION.  
SW<06> = 1 ..... OPTIONAL TIMEOUT OF RETRY ERRORS  
SW<05> = 1 ..... INHIBIT PASS COUNT  
SW<04> = 1 ..... ALLOWS 8 ERROR TIMEOUTS IN THE  
COMPARE ROUTINE BEFORE EXECUTING NEXT READ  
COMMAND. WHEN SWITCH IS 0, ONLY 1 ERROR  
TIMEOUT IS RECORDED.  
SW<03> = 1 ..... TIMEOUT # OF ERRORS  
SW<02> = 1 ..... INHIBIT MEMORY MANAGEMENT  
SW<01> = 1 ..... DATA TEST ONLY  
SW<00> = 1 ..... DROPS DRIVE AFTER 20 ERRORS

### 5.1 Data Reliability Test Mode

With SW8 set, the program will set the 'BAI' bit in RHCS2 and transfer 64K of data at a time for all patterns except random. Random will be executed as usual with standard buffers. No compares are done in this mode of operation except on random patterns. This option should only be used in data test or conversation mode. When used in conversation mode it overrides the non standard word count. You should not select a desired disk address in conversation mode for it can produce a disk address overflow error for this data reliability test mode only does 64K word transfers. If SW8 is changed, while the program is running, the program will finish its pass before executing the switch change.

### 5.2 Conversation Mode for Program Parameters for Data Test Only

In conversation mode the operator can specify any one or all of the program parameters.

#### NOTE

Once in conversation mode, the only way to remap the system is to reload the program. To restart the program in conversation mode without having to reanswer the questions, the starting address is 210. Reset switch 10. To restart the program without having to reanswer the port sizing questions, restart at 220. Reset switch 10.

The program will now ask several questions, the table below will help you answer the questions.

TYPE TO START AT		TYPE TO START AT	
0	000000		
1	020000	20	400000
2	040000	21	420000
3	060000	22	440000
4	100000	23	460000
5	120000	24	500000
6	140000	25	520000
7	160000	26	540000
10	200000	27	560000
11	220000	30	600000
12	240000	31	620000
13	260000	32	640000
14	300000	33	660000
15	320000	34	700000
16	340000	35	720000
17	360000	36	740000
		37	760000

TYPE TO START AT		TYPE TO START AT	
40	1000000	60	1400000
41	1020000	61	14200000
42	1040000	62	1440000
43	1060000	63	1460000
44	1100000	64	1500000
45	1120000	65	1520000
46	1140000	66	1540000
47	1160000	67	1560000
50	1200000	70	1600000
51	1220000	71	1620000
52	1240000	72	1640000
53	1260000	73	1660000
54	1300000	74	1700000
55	1320000	75	1720000
56	1340000	76	1740000
57	1360000	77	1760000

TYPE TO START AT		TYPE TO START AT	
100	2000000	120	2500000
101	2020000	121	25200000
102	2040000	122	2540000
103	2060000	123	2560000
104	2200000	124	2600000
105	2220000	125	2620000

106	2240000	126	2640000
107	2260000	127	2660000
110	2300000	130	2700000
111	2320000	131	2720000
112	2340000	132	2740000
113	2360000	133	2760000
114	2400000	134	3000000
115	2420000	135	3020000
116	2440000	136	3040000
117	2460000	137	3060000

NOTE: The formula to get numbers to be typed is: Every 4K boundary end in four zeros so disregard the last four digits and divid the remaining address by two. The resulting number is to be typed in for that 4K bank.

NOTE: Type only numbers shown!!!

1. Type starting 4K bank # for data buffer on port A

This will determine where your buffer area will start on -A- port. Use table above

NOTE:

Program is located in 1st 4K bank. Therefore, this bank can not be used as a buffer.



EXAMPLE:

```

XXXXXXXXXX 16k
X          X
X  BANK 3  X
X          X
XXXXXXXXXX 12k
X          X
X  BANK 2  X
X          X
XXXXXXXXXX 8k
X          X
X  BANK 1  X
X          X
XXXXXXXXXX 4k
X          X
X  BANK 0  X
X PROGRAM X
XXXXXXXXXX 0

XXXXXXXXXX
X          X
X (CPU X    8k XXXXX          XXXXXX
X          X    XMEMX        X RH X
XXXXXXXXXX 0 XXXXX          XXXXXX
X          X          X
X          X          X-A-PORT
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

These answers given below will test the configuration in the given example. Answers:

- To Test -A- port
- 1) 1
  - 2) 1

Program Conversation

MULTI DRIVE MODE? (YES-NO)

Multi disk mode is a mode in the program which allows the operator to exercise all the disks on the system without re-starting the program. The program, after exercising one disk will report a message telling the operator which disk will be selected next, and then the program will exercise that disk. When a complete pass is accomplished, a pass complete will be reported and the test will recycle.

If the answer to the multi drive mode was 'NO', the following question is asked.

TYPE UNIT #

The operator can now select the unit he wishes to test by typing the unit number.

OPTIONAL WORD COUNT (YES-NO)

If the operator answers 'NO' to this question the next question will be deleted from the conversation.

WD CT

The operator can specify any length transfer from 1(8). The normal transfer length is n(8) words where n is the maximum buffer size for the available core.

This program maps the system in 4K segments. If there is a 1K block of memory on the system that you would like to reach, you can type in that 4K bank # and then specify a wc of 2000.

If the word count number typed, is larger than the core size given in the setup routine, the question will be repeated.

OPTIONAL DSK ADDR (YES-NO)

If the answer to this question is no, the whole disk will be written and the next question is not asked.

DSK ADDR

The operator can now specify the starting sector

DATA PATTERN NO.?

If no optional data pattern is requested (#22) the program will execute the following list of data patterns.

PATTERN	0 = 000000
..	1 177777
..	2 031463
..	3 = 066666
..	4 100001
..	5 = 107070
..	6 = 070707
..	7 = 052525
..	10 125252
..	11 177737
..	12 146314
..	13 = 136363
..	14 = 063636
..	15 = 000001
..	16 = 100005
..	17 = 155555
..	20 = 133333
..	21 = Random data
..	22 = Run all data patterns under program control

In this section of the program parameter conversation mode, the operator can select any one or all three of the control functions to be executed. The normal sequence of disk functions under program control are WRITE, WRITE CHECK, and then READ. By entering the conversation mode the operator has gained complete control over the disk functions. He must specify yes or no to all of the following questions.

WRITE? (YES - NO)  
READ? (YES - NO)  
WRITE CHECK? (YES - NO)

To perform a WRITE CHECK only, the operator must first write some known data on the disk. This course of action also prevails for a READ only operation.

\* If an error occurs in the line the operator is typing, depress the rub-out key and retype answer.

ALL ANSWERS SHOULD BE FOLLOWED BY A CARRIAGE-RETURN

5.3 Routine Abstracts

ADDRESS TEST

This test writes each sector with its own address then reads it back and compares it for the correct data.

#### RANEX - Random data, address and word count test

This routine tests the ability of the system to access random addresses with random data. Two sectors of random data are written at a starting random address on the disk. It is then write checked and read. All errors are reported. This is repeated 1000 times.

#### DATA RELIABILITY - data pattern test

In this portion of the test, the reliability of the disk surface is tested by WRITE, WRITE CHECK, and READ functions. The routine first writes the complete surface with a set data pattern, then a write check of the complete surface is accomplished, thus reporting all errors between the data written and the data in memory. The disk is then read. The data read from the disk is compared against the known data pattern. This compare is taking place the same time the disk is being read. The buffer is cleared as it is being compared.

#### 5.4 Subroutine Abstracts

##### 5.4.1 SCOPE

This subroutine call is placed between each subtest in the instruction section. It records the starting address of each subtest as it is being entered in location 'LAD'. If a scope loop is requested, the current subtest will be looped upon. The contents of LAD may be used to determine the last subtest successfully completed.

##### 5.4.2 HLT

This routine prints out an error message (See 6.1). To inhibit typeouts, put SW<13> on a 1.

##### 5.4.3 TRAPCATCHER

A '+2' - 'HALT' sequence is repeated from 0 - 776 to catch any unexpected traps. Thus any unexpected traps or interrupts will HALT at the vector + 2.

## 6. ERRORS

### 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.

If SW0 is set, a drive will be dropped from the test sequence after 20 errors. The program will state which drive was dropped and on which pass it was dropped. If all the drives have been dropped, the program will type 'TESTING UNIT 0' and HALT', indicating that it could not find any more drives on the system to test.

## 7. RESTRICTIONS

None

## 8. MISCELLANEOUS

### 8.1 Execution Time

Pass complete will be typed out at end of pass. It will take between 15 to 20 minutes to complete a pass. add 30 seconds for each 4K. for data test.

### 8.2 Stack Pointer

Stack is initially set to 500

### 8.3 Power Fail

The starting address for the Write Power Fail Test is 270. A message will be typed out "load SW with unit # and cont." The operator now has to load the unit # in octal into the SW register in bits 00-01 and 02. Then hit continue. The program will tell the operator when to power down. When the system is powered up, only one error is allowed. The starting address for the Writecheck Power Fail Test is 274. Here as in the Write Power Fail Test, the program will tell the operator when to power down. When the power comes back, no errors should occur.

.TITLE CERSB-C RH70-RS03 DATA AND RELIABILITY TEST  
 :COPYRIGHT 1973,1974,1975, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.  
 :PROGRAM BY STANLEY HARACKIEWICZ

			SWITCH	USE
1				
2				
3				
4				
5				
6				
7			SW15= 10000	:HALT ON ERROR
8			SW14= 40000	:LOOP ON FUNCTION
9			SW13= 20000	:INHIBIT ERROR TYPEOUTS
10			SW12= 10000	:INHIBIT COMPARISON
11			SW11= 4000	:HALT ON COMPLETION OF TRANSFER
12			SW10= 2000	:CONVERSATION MODE
13			SW9= 1000	:LOOP ON ERROR
14			SW8= 400	:
15			SW7= 200	:WAIT IN BACKGROUND TEST
16			SW6= 100	:OPTIONAL TYPEOUT OF RETRY ERRORS
17			SW5= 40	:INHIBIT PASS COUNT AND UNIT #
18			SW4= 20	:ALLOWS 8 LOCATIONS TO BE TESTED IN COMPARE ROUTINE
19			SW3= 10	:TYPE OUT TOTAL # OF ERRORS
20			SW2= 4	:INHIBIT MEMORY MANAGEMENT
21			SW1= 2	:DATA TEST ONLY
22			SW0= 1	:DROP DRIVE AFTER 20 ERRORS
23			0	:TRAP CATCHER FROM 0 - 776
24			46	:HOOKS FOR ACT 11
25	000046	013776	SENDAD	
26		000052		
27	000052	040000	BIT14	
28				
29		000200	200	
30	000200	000137 001234	JMP @WBEGIN	:START TEST
31				
32		000210	210	
33	000210	012706 000500	MOV #500,SP	:SETUP STACK
34	000214	000137 003176	JMP @WADTST	:RESTART ADDR
35				
36		000220	220	
37	000220	012706 000500	MOV #500,SP	:CONVERSATION MODE WITHOUT
38	000224	000137 002322	JMP @WA1	:DATA BUFFER QUESTIONS
39				
40		000230	230	
41	000230	000137 015142	JMP @WRLLDR	:RESTORE LOADER
42				
43		000260	260	
44	000260	000137 003252	JMP @WADTL	:TRACK AND SECTOR SELECT TEST
45				:WRITE EACH WORD ADDR ON ITSELF AND READ IT BACK
46				:LOCATION 1150 CONTAINS UNIT NO.
47		000264	264	
48	000264	000137 005130	JMP @WRANEL	:RANDOM ADDRESS, DATA TEST
49				:LOCATION 1150 CONTAINS UNIT NO.
50		000270	270	
51	000270	000137 012334	JMP @WPF11	:DISK WRITE POWER FAIL TEST
52		000274	274	
53	000274	000137 012670	JMP @WPF12	:DISK WRITE CHECK POWER FAIL TEST

```
54 ;RH70 DATA PATTERNS
55
56 000300 000000 PAT0: 0
57 000302 177777 PAT1: 177777
58 000304 031463 PAT2: 031463
59 000306 066666 PAT3: 066666
60 000310 100001 PAT4: 100001
61 000312 107070 PAT5: 107070
62 000314 070707 PAT6: 070707
63 000316 052525 PAT7: 052525
64 000320 125252 PAT10: 125252
65 000322 177737 PAT11: 177737
66 000324 146314 PAT12: 146314
67 000326 136363 PAT13: 136363
68 000330 063636 PAT14: 063636
69 000332 000001 PAT15: 000001
70 000334 100005 PAT16: 100005
71 000336 155555 PAT17: 155555
72 000340 133333 PAT20: 133333
73 ;PAT21 RANDOM DATA
74
75 ;CLEAR ALL REGISTERS
76 000342 012777 000040 000464 .CLR DV: MOV #40,@RSCS2 ;CLEAR ALL REG
77 000350 013777 001164 000456 MOV UNNUM,@RSCS2 ;GET UNIT #
78 000356 000002 RTI
```



```
.SBTTL          SKMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

79
80
81          177572          SR0=177572          ;ADDRESS OF MEM MGMT REGISTER SR0
82          177574          SR1=177574          ;          ..          ..          ..          SR1
83          177576          SR2=177576          ;          ..          ..          ..          SR2
84          172516          SR3=172516          ;ADDRESS OF MEM MGMT REGISTER SR3
85
86          172300          KIPDR0=172300          ;ADDRESS OF KERNEL 'I' PAGE
87          172302          KIPDR1=172302          ;DESCRIPTOR REGISTERS
88          172304          KIPDR2=172304
89          172306          KIPDR3=172306
90          172310          KIPDR4=172310
91          172312          KIPDR5=172312
92          172314          KIPDR6=172314
93          172316          KIPDR7=172316
94
95          172320          KDPDR0=172320          ;ADDRESSES OF KERNEL 'D' PAGE
96          172322          KDPDR1=172322          ;DESCRIPTOR REGISTERS
97          172324          KDPDR2=172324
98          172326          KDPDR3=172326
99          172330          KDPDR4=172330
100         172332          KDPDR5=172332
101         172334          KDPDR6=172334
102         172336          KDPDR7=172336
103
104         172340          KIPAR0=172340          ;ADDRESSES OF KERNEL 'I' PAGE
105         172342          KIPAR1=172342          ;ADDRESS REGISTERS
106         172344          KIPAR2=172344
107         172346          KIPAR3=172346
108         172350          KIPAR4=172350
109         172352          KIPAR5=172352
110         172354          KIPAR6=172354
111         172356          KIPAR7=172356
112
113         172360          KDPAR0=172360          ;ADDRESSES OF KERNEL 'D' PAGE
114         172362          KDPAR1=172362          ;ADDRESS REGISTERS
115         172364          KDPAR2=172364
116         172366          KDPAR3=172366
117         172370          KDPAR4=172370
118         172372          KDPAR5=172372
119         172374          KDPAR6=172374
120         172376          KDPAR7=172376
```

121	000001	N=	1	:INITALIZE FOR NEWTST
122	104000	HLT-	EMT	:SET HLT TO EMT FOR ERROR TYPEOUTS
123	177776	PS=	177776	:PROCESSOR STATUS
124	177776	PSW=	PS	:PROCESSOR STATUS WORD
125	177570	SWR=	177570	:SWITCH REGISTER
126	177570	DISPLAY=	SWR	:DISPLAY REGISTER
127	000007	BELL=	7	:BELL
128	000000	R0=	%0	:R0 - DEFINE REGISTERS
129	000001	R1=	%1	:R1
130	000002	R2=	%2	:R2
131	000003	R3=	%3	:R3
132	000004	R4=	%4	:R4
133	000005	R5=	%5	:R5
134	000006	SP=	%6	:R6 - STACK POINTER
135	000007	PC=	%7	:R7 - PROGRAM COUNTER
136	000001	BIT0=	1	:BIT EQUATES
137	000002	BIT1=	2	
138	000004	BIT2=	4	
139	000010	BIT3=	10	
140	000020	BIT4=	20	
141	000040	BIT5=	40	
142	000100	BIT6=	100	
143	000200	BIT7=	200	
144	000400	BIT8=	400	
145	001000	BIT9=	1000	
146	002000	BIT10=	2000	
147	004000	BIT11=	4000	
148	010000	BIT12=	10000	
149	020000	BIT13=	20000	
150	040000	BIT14=	40000	
151	100000	BIT15=	100000	
152	000001	GOOD=	R1	:FOR GOOD DATA
153	000000	BAD=	R0	:FOR BAD DATA
154	000000			



206 001000 .- 1000  
 207  
 208 001000 000000  
 209 001002 000000  
 210 001004 000000 000000  
 211 001010 000000  
 212 001012 000000  
 213 001014 001000  
 214 001016 177564  
 215 001020 177566  
 216  
 217 001022 000000  
 218 001024 000000  
 219 001026 000000  
 220 001030 000000  
 221  
 222  
 223  
 224  
 225  
 226

ICNT: 0 ;LH = ITERATION COUNT ;RH = TEST NO.  
 ERRORS: 0 ;ERROR COUNT  
 PCNT: 0,0 ;2 WORD PASS COUNT  
 LAD: 0 ;LOOP ADDRESS FOR SCOPE  
 HLTADR: 0 ;ADDRESS OF LAST HLT INSTRUCTION EXECUTED  
 FILCHR: 1000 ;FILCHR=0 (CHAR) ;FILCHR+1=2 (COUNT)  
 TPS: 177564 ;OUTPUT STATUS REGISTER  
 TPB: 177566 ;OUTPUT BUFFER  
 TEMP1: 0  
 TEMP2: 0  
 TEMP3: 0  
 TEMP4: 0

:DISK I/O REGISTERS

227 001032 172040  
 228 001034 172050  
 229 001036 172042  
 230 001040 172044  
 231 001042 172046  
 232 001044 172052  
 233 001046 172054  
 234 001050 172056  
 235 001052 172060  
 236 001054 172062  
 237 001056 172064  
 238 001060 172066  
 239 001062 172070  
 240 001064 172072  
 241 001066 000204  
 242  
 243  
 244  
 245  
 246  
 247  
 248  
 249  
 250  
 251  
 252  
 253  
 254  
 255 001070 000206  
 256 001072 000200

RSCS1: 172040 ;DISK CONTROL + STATUS REGISTER  
 RSCS2: 172050 ;DISK CONTROL + STATUS REGISTER  
 RSWC: 172042 ;WORD COUNT REGISTER  
 RSBA: 172044 ;BUS ADDRESS  
 RSDA: 172046 ;DISK ADDRESS (DESIRED ADDRESS)  
 RSDS: 172052 ;DRIVE STATUS  
 RSER: 172054 ;ERROR REG.  
 RSAS: 172056 ;ATTENTION SUMMARY  
 RSLA: 172060 ;LOOK AHEAD  
 RSDB: 172062 ;DATA BUFFER REGISTER  
 RSMR: 172064 ;MAINTENANCE REGISTER  
 RSDT: 172066 ;DRIVE TYPE REGISTER  
 RSBAE: 172070 ;BUS ADDRESS EXTENSION  
 RSCS3: 172072 ;CONTROL AND STATUS 3  
 RSVEC: 204 ;INTERUPT RSVEC

:BIT ASSIGNMENTS FOR ERROR TYPE OUTS

DB=2  
 DA=4  
 WC=10  
 BA=20  
 DS=40  
 AS=100  
 LA=204  
 MR=220  
 DT=240  
 STATUS: 206  
 PRIORITY:BIT7

;DATA BUFFER  
 ;DESIRED ADD  
 ;WORD COUNT  
 ;BUS ADDRESS  
 ;DRIVE STATUS  
 ;ATTENTION SUMMARY  
 ;LOOK AHEAD  
 ;MAINTENANCE  
 ;DRIVE TYPE  
 ;DISK INTERRUPT STATUS  
 ;DISK PRIORITY LEVEL

257	000006	RW=6	;R/W IN PDR REG
258	000000	UP=0	;UP BITY IN PDR REG
259	000250	MMVEC=250	;ADDR OF MEM MGMT ERROR TRAP
260	001074	STAMEM: 0	;STARTING LOC FOR -A- PORT
261	001076	SAVAST: 0	;SAVE LOC FOR STAMEM
262	001100	STBCOM: 0	;STARTING LOC FOR -B- PORT
263	001102	SAVCPU: 0	;SAVE LOC FOR CPUEM
264	001104	SAVMGA: 0	;STARTING ADDR FOR -A- PORT WITH MEM MGMT
265	001106	SAVMGB: 0	;STARTING ADDR FOR B PORT W/MEM MGMT
266	001110	SAVMGC: 0	;STARTING LOC FOR CPU W/MEM MGMT
267	001112	SIZEAP: 0	;SIZE OF A PORT
268	001114	SIZEBP: 0	;SIZE OF B PORT
269	001116	WDCTB: 0	;WC FOR A PORT
270	001120	AOB1: 0	;FLAG FOR PORT BEING TESTED
271	001122	VADDR: 0	;VIRTUAL ADDR
272	001124	PHADDR: 0	;PHYSICAL ADDR
273	001126	FLAG2: 0	;FLAG FOR RESTART AND FOUND DRIVE
274	001130	DROP: 0	;BAD UNITS ON SYSTEM THAT GET DUMPED

:DISCRIPTION OF FLAG2

:BIT0 = RESTART  
 :BIT1 = FOUND DRIVE  
 :BIT2 = ERROR DO A CRLF FOR UNIT #  
 :BIT3 = DOING COMPARE  
 :BIT4 = SET A16 IN CS1  
 :BIT5 = SET A17 IN CS1  
 :BIT6 = SET IF MEMORY HAS ALREADY BEEN FOUND  
 :BIT7 = WHEN SET MAKE WC UP TO 28K  
 :BIT8 = FOUND MEMORY ON -B- PORT  
 :BIT9 = POWER DID FAIL  
 :BIT10 = WAITING IN BACKGROUND TEST  
 :BIT11 = PARITY ERROR ROUTINE  
 :BIT12 = POWER FAIL TEST  
 :BIT13 = IN MAP ROUTINE  
 :BIT14 = IN POWER FAIL OR CONVERSATION MODE  
 :BIT15 = ERROR IN POWER FAIL

:DISCRIPTION OF FLAG

:BIT0 = USED FOR WRITE COUNTER  
 :BIT1 = USED FOR WRITE COUNTER  
 :BIT2 = TRANSFER MODE 64K  
 :BIT5 = OPTIONAL DMA  
 :BIT6 = TEST -B- PORT  
 :BIT7 = LAST DISK BUFFER FLAG  
 :BIT8 = PROGRAM IS IN ADDRESS OR RANDOM TEST  
 :BIT9 = ERROR DURING TRANSFER  
 :BIT10 = DATA TEST ONLY  
 :BIT11 = MULTIPOINT  
 :BIT12 = READ  
 :BIT13 = WRITE CHECK  
 :BIT14 = WRITE  
 :BIT15 = PROGRAM CONTROL MODE

257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310

```

311          :RH70 DEDICATE REGISTERS (MEMORY)
312
313 001132 000000 FLAG: 0          :TEST REGISTER
314 001134 000000 WRDCT: 0       :WORKING WORD COUNT
315 001136 000000 TRACK: 0      :WORKING DAE
316 001140 000000 DMA: 0        :WORKING DAR
317 001142 000000 PATNU: 0     :DATA PATTERN INDEX
318 001144 000000 BUF: 0       :WORKING DATA BUFFER (OUT-IN)
319 001146 000000 TDMA: 0      :TEMP DAR
320 001150 000000 SWRDCT: 0    :STANDARD WORD COUNT
321 001152 000000 ERCOUNT: 0  :ERROR COUNT FOR MESSAGES.
322 001154 000000 SAVE: 0
323 001156 000000 HRDR: 0          :POINTER FOR HARD ERROR
324 001160 000000 BLOCK: 0
325 001162 000000 PASSC: 0
326 001164 000000 UNNUM: 0        :UNIT CURRENTLY BEING TESTED
327 001166 000000 UNITSV: 0       :SET BIT=UNIT ON BUS
328 001170 000000 UNCMP: 0       :FOR COMPARING FOR # OF DEVICE
329 001172 000000 RS04DT: 0     :FLAG FOR RS04
330 001174 000000 NUMS: 0        :WORK LOC FOR NUMBER INPUTS
331 001176 000000 CMD: 0         :LOC FOR CS2 COMMANDS
332 001200 000000 SWITCH: 0     :FLAG FOR WHICH RANDOM NUMBER GEN
333 001202 000000 INTFLG: 0    :FLAG FOR INTERRUPT
334 001204 000000 LOPCNT: 0     :ERROR FLAG AND LOOP COUNTER FLAG
335 001206 000000 WRITER: 0    :CONTAINS # OF WRITE ERRORS
336 001210 000000 WCERR: 0     :CONTAINS # OF WRITE CHECK ERRORS
337 001212 000000 READER: 0    :CONTAINS # OF READ ERRORS
338 001214 000000 COMERR: 0    :CONTAINS # OF COMPARE ERRORS
339 001216 000000 MMAVA: 0     :MEM MGMT AVAILABLE INDICATOR
340 001220 000000 SAVWC: 0     :SAVE LOC FOR CONVERSATION WC ROUTINE
341 001222 000000 FLAG3: 0     :LOOP IN ADDRESS + RANDOM TST FLAG
342 001224 000000 SAVWCB: 0    :SAVE WC SIZE FOR -B- PORT
343
344          :RH70 WORK REGISTERS
345          :((CAN BE CHANGED IN ANY ROUTINE))
346 001226 000000 WORK: 0
347 001230 000000 WORK1: 0
348 001232 000000 WORK2: 0
349          ERRVEC=4
350          LERADD=177740
351          HERADD=177742
352          MEMERR=177744
  
```

353	001234	012706	000500		BEGIN:	MOV	#500,SP	:SET STACK TO *** 500 ***
354	001240	012737	016350	000024		MOV	#.POWER,@#24	:SET UP PF VECTOR
355	001246	012737	000340	000026		MOV	#340,@#26	:LOCK OUT THE WORLD
356	001254	012737	016014	000030		MOV	#.HLT,@#30	:SET EMT VECTOR
357	001262	012737	000340	000032		MOV	#340,@#32	:LOCK UP
358	001270	012737	016744	000034		MOV	#.TRAP,@#34	:SET TRAP VECTOR
359	001276	012737	000340	000036		MOV	#340,@#36	:LOCK UP
360	001304	005037	001000			CLR	ICNT	:INIT ICNT
361	001310	005037	001010			CLR	LAD	:INIT LAD
362	001314	042737	177677	001132		BIC	#177677,FLAG	:CLEAR FLAG
363	001322	042737	177776	001126		BIC	#177776,FLAG2	:CLEAR ALL EXECPT RESTART
364	001330	005037	001222			CLR	FLAG3	:CLEAR LOOP IN ADDRESS + RANDOM TST FLAG
365	001334	032737	000001	001126		BIT	#BIT0,FLAG2	:IS THIS THE FIRST TIME?
366	001342	001002				BNE	1\$	:NO
367	001344	004737	020000			JSR	PC,LDR	:SAVE LOADER
368	001350	000005			1\$:	RESET		:CLEAR THE WORLD
369	001352	012737	000340	177776		MOV	#340,PS	:LOCK UP INTERRUPT LEVELS
370	001360	004537	012300			JSR	R5,ERRCL	:CLEAR ERROR COUNTER + PASS CNT
371	001364	005037	001216			CLR	MMAVA	:CLEAR MEM MGMT FLAG
372	001370	005037	001120			CLR	AOB1	:TEST A PORT FIRST
373	001374	032737	000004	177570		BIT	#BIT2,SWR	:WANT MEM MGMT?
374	001402	001021				BNE	3\$	:NO
375	001404	012737	001432	000004		MOV	#5\$,4	:SET TIMEOUT TRAP
376	001412	012737	000340	000006		MOV	#340,6	:SET PS
377	001420	005037	177572			CLR	@MSR0	:IS MEM MGMT AVAILABLE?
378	001424	005137	001216			COM	MMAVA	:YES
379	001430	000401				BR	4\$	:CONT
380	001432	022626			5\$:	CMP	(6)+,(6)+	:CLEAR STACK
381	001434	012737	000006	000004		4\$:	MOV	#6,4
382	001442	005037	000006			CLR	6	:TRAP
383	001446	032737	000001	001126		3\$:	BIT	#BIT0,FLAG2
384	001454	001002				BNE	CALM	:IS THIS THE FIRST TIME
385	001456	000137	020070			JMP	SIZZAP	:NO
386	001462	004737	011464		CALM:	JSR	PC,@EXTMEM	:SIZE BUFFERS
387	001466	004737	015170		CALM1:	JSR	PC,.MAMK	:SET UP DATA BUFFERS
388	001472	032737	000001	001126		BIT	#BIT0,FLAG2	:TURN ON PARITY MEM
389	001500	001003				BNE	3\$	:1ST TIME ?
390	001502	013737	001150	001220		MOV	SWRDCT,SAVWC	:NO
391	001510	052737	000001	001126		3\$:	BIS	#BIT0,FLAG2
392	001516	005037	001140			CLR	DMA	:SAVE WC FOR CONVERSATION MODE COMPARE
393	001522	005037	001142			CLR	PATNU	:SET 1ST TIME FLAG
394	001526	013737	001150	001134		MOV	SWRDCT,WRDCT	:CLEAR DAR REGISTERS
395	001534	032737	000002	177570		BIT	#BIT1,SWR	:CLEAR PATTEN COUNT
396	001542	001403				BEQ	2\$	:DATA TEST ONLY?
397	001544	052737	002000	001132		BIS	#BIT10,FLAG	:NO
398	001552	032737	002000	177570		2\$:	BIT	:YES
399	001560	001015				BNE	1\$	:ENTER CONVERSATION MODE?
400	001562	052737	074000	001132		BIS	#74000,FLAG	:YES GO TO CONVERSATION MODE
401	001570	004537	010214			JSR	R5,RESTOR	:RESTORE ORIGINAL WD CNT
402	001574	012737	017426	000004		MOV	#TIEOUT,ERRVEC	
403	001602	012737	000340	000006		MOV	#340,ERRVEC+2	
404	001610	001137	003176			JMP	ADTST	
405	001614	000137	002220		1\$:	JMP	@CONM	:ENTER CONVERSATION MODE

```

406                                     :FIND OUT HOW MANY DRIVES
407                                     :FIRST TEST RSAS
408
409 001620 012701 000010                DRVENO: MOV      #8,R1          :PUT 8 INTO R1 FOR COUNT
410 001624 042737 000002 001126        BIC      #BIT1,FLAG2        :CLEAR FOUND DRIVE FLG
411 001632 012777 000000 177174        MOV      #0,@RSCS2         :SET DEVICE TO ZERO
412 001640 012777 000007 177200        TRY:    MOV      #7,@RSER   :CAUSE AN ERROR +SETS BIT IN AS REG
413 001646 005301                                DEC      R1                :DO A MAXIMUM OF 16 TIMES
414 001650 001403                                BEQ     DVNUM              :TESTED FOR ALL DRIVES GET OUT
415 001652 005277 177156                INC     @RSCS2            :INCREMENT DRIVE UNIT
416 001656 000770                                BR      TRY                :REPEAT FOR NEXT DRIVE
417 001660 017737 177164 001166        DVNUM: MOV     @RSAS,UNITSV :SAVE
418 001666 043737 001130 001166        BIC     DROP,UNITSV       :DROP BAD DRIVES
419 001674 012737 000401 001170        MOV     #401,UNCOMP       :SETUP TO CMP WITH UNITSV
420 001702 012737 000000 001164        MOV     #0,UNNUM          :PUT 0 INTO UNIT NO.
421 001710 032737 000040 177570        BIT     #BIT5,SWR         :INHIBIT TYPE OUT?
422 001716 001015                                BNE     STTEST            :YES
423 001720 104402 001724                                TYPE   ,+2                :.ASCIZ <15><12>'TESTING UNIT ''
424 001744 042737 000004 001126        BIC     #BIT2,FLAG2        :CLEAR ERROR FLAG
425 001752 033737 001170 001166        STTEST: BIT    UNCOMP,UNITSV :IS THIS DRIVE ON THE SYSTEM
426 001760 001463                                BEQ     TRYNX              :NO
427 001762 013777 001164 177044        UNTP:  MOV     UNNUM,@RSCS2 :YES PUT UNIT # INTO CS2
428 001770 005037 001172                                CLR     RS04DT            :CLEAR DRIVE TYPE FLAG
429 001774 005777 177060                                TST    @RSDT              :IS THIS A RS03?
430 002000 001417                                BEQ     1$                 :YES
431 002002 022777 000001 177050        2$:    CMP     #1,@RSDT     :IS THIS A RS03 4US?
432 002010 001413                                BEQ     1$                 :YES
433 002012 022777 000002 177040        3$:    CMP     #2,@RSDT     :IS THIS A RS04?
434 002020 001404                                BEQ     6$                 :YES
435 002022 022777 000003 177030        CMP     #3,@RSDT         :RS04?
436 002030 001037                                BNE     TRYNX              :GET A NEW NUMBER
437 002032 052737 177777 001172        6$:    BIS     #-1,RS04DT   :YES RS04
438 002040 032737 040000 001126        1$:    BIT     #BIT14,FLAG2 :IN POWER FAIL OR CONVERSATION?
439 002046 001401                                BEQ     7$                 :NO
440 002050 000207                                RTS     PC                 :YES
441 002052 032777 000200 176764        7$:    BIT     #BIT7,@RSDS  :IS THIS DRIVE READY ?
442 002060 001423                                BEQ     TRYNX              :NO GET ANOTHER DRIVE
443 002062 032737 000040 177570        BIT     #BIT5,SWR         :TYPEOUT?
444 002070 001016                                BNE     4$                 :NO
445 002072 032737 000004 001126        BIT     #BIT2,FLAG2       :WAS THERE AN ERRER?
446 002100 001402                                BEQ     5$                 :NO
447 002102 104402 000757                                TYPE   ,CRLF              :
448 002106                                5$:
449 002106 013746 001164                MOV     UNNUM,-(6)         :PUT UNNUM ON STACK
450 002112 104406                                TYPES  :TYPE STACK IN OCTAL - SUPRESS
451 002114 104402 000040                TYPE   ,40                :TYPE SPACE
452 002120 042737 000004 001126        BIC     #BIT2,FLAG2        :CLEAR ERROR FLAG
453 002126 000430                4$:    BR      NOWGO         :NOW TEST

```



```

454 002130 006337 001170      TRYNX:  ASL      UNCMP      ;CHECK NEXT BIT FOR DRIVE
455 002134 103403              BCS      CHCKDV     ;DID WE TEST ANY REG?
456 002136 005237 001164      INC      UNNUM      ;INC UNIT #
457 002142 000703              BR       STTEST     ;CHECK FOR NEXT DRIVE
458
459 002144 032737 000002 001126  CHCKDV:  BIT      #BIT1,FLAG2 ;FOUND DRIVE?
460 002152 001014              BNE      DONEE      ;YES WE DID TEST A DRIVE
461 002154 012737 100000 001170  MOV      #100000,UNCMP ;NO DRIVES TESTED, COULD NOT SET
462 002162 005037 001164      CLR      UNNUM      ;ANY AS BITS, THUS DEFAULTS TO 0
463 002166 013746 001164      MOV      UNNUM,-(6) ;PUT UNNUM ON STACK
464 002172 104406              TYPES    ;TYPE STACK IN OCTAL - SUPRESS
465 002174 104402 000730      TYPE    .NOFND
466 002200 000000              HALT      ;COULD NOT SET ANY ATA BITS
467                                     ;BY SETTING ERROR BITS
468                                     ;GO BACK AND USE OTHER DIAG.
469 002202 000402              BR       NOWGO      ;TEST DRIVE 0
470 002204 000137 013344      DONEE:  JMP      OUT      ;GET OUT
471 002210 052737 000002 001126  NOWGO:  BIS      #BIT1,FLAG2 ;FOUND DRIVE
472 002216 000207      RTS      PC
473
474                                     ;ENTER OPERATOR CONVERSATION MODE
475
476 002220 104402 006566      CONM:   TYPE    .STABUF
477 002224 104420              RDOCT
478 002226 012637 001074      MOV      (6)+,STAMEM ;START BUFFER AT 4K
479 002232 000137 002310      JMP      NOPORT     ;GET OUT
  
```

480	002236	104402	006566		1\$:	TYPE	,STABUF		
481	002242	104402	002246			TYPE	..+2		:.ASCIZ 'B ''
482	002252	104420				RDOCT			:GET ANS
483	002254	012637	001100			MOV	(6)+,STBCOM		:AND SAVE IT
484	002260	104402	006613		4\$:	TYPE	,BUFSIZ		
485	002264	104420				RDOCT			:GET ANS
486	002266	012637	001114			MOV	(6)+,SIZEBP		:SAVE IT
487	002272	022737	000006	001114		CMP	#6,SIZEBP		:GREATER THEN 24K?
488	002300	002767				BLT	4\$		:YES ASK AGAIN
489	002302	052737	000100	001132		BIS	#BIT6,FLAG		:SET B PORT FLAG
490	002310	004737	011464		NOPORT:	JSR	PC,EXTMEM		:CAL BUFFERS AND WC
491	002314	013737	001150	001134		MOV	SWRDCI,WRDCT		:GET STANDARD WC
492	002322	052737	002000	001132	A1:	BIS	#BIT10,FLAG		:SET BIT FOR DATA TEST ONLY
493	002330	004537	012300			JSR	R5,ERRCL		:CLEAR ERROR CNT + PASS CNT
494	002334	042737	174040	001132		BIC	#174040,FLAG		:CLEAR MULTI FLAG MODE +PATTERN SELECT
495	002342	104402	002346			TYPE	..+2		:.ASCIZ <15><12>'MULTI DRIVE''
496	002364	004737	003234			JSR	PC,CMPY		:COMPARE FOR YES
497	002370	001004				BNE	DATTES		:ANS IS NO
498	002372	052737	004000	001132		BIS	#BIT11,FLAG		:SET BIT FOR MULTI DRIVE
499	002400	000444			1\$:	BR	ASKWC		
500	002402				DATTES:				
501	002402	104402	002406			TYPE	..+2		:.ASCIZ <15><12>'TYPE UNIT # ''
502	002426	104420				RDOCT			
503	002430	012637	001174			MOV	(6)+,NUMS		:GET NUMBER
504	002434	022737	000010	001174		CMP	#10,NUMS		:CORRECT # ?
505	002442	103757				BLO	DATTES		:NO
506	002444	013737	001174	001164		MOV	NUMS,UNNUM		:SET UNIT #
507	002452	004737	006740			JSR	PC,FNDTYP		:TEST FOR RS04 OR 03
508	002456	005002			1\$:	CLR	R2		:CLEAR WORK AREA
509	002460	000261				SEC			:SET CARRY
510	002462	006102			2\$:	ROL	R2		:SET BIT IN WORK
511	002464	005737	001174			TST	NUMS		:IS THIS THE RIGHT BIT FOR THE RIGHT DISK
512	002470	001403				BEQ	3\$		:YES
513	002472	005337	001174			DEC	NUMS		:NO TRY AGAIN
514	002476	000771				BR	2\$		:TEST AGAIN
515	002500	010237	001166		3\$:	MOV	R2,UNITSV		:SET DRIVE BIT IN UNITSV
516	002504	052737	000002	001126		BIS	#BIT1,FLAG2		:SET FOUND DRIVE FLAG
517									
518	002512				ASKWC:				
519	002512	104402	002516			TYPE	..+2		:.ASCIZ <15><12>'OPTIONAL WD CT''
520	002540	004737	003234			JSR	PC,CMPY		:COMPARE FOR YES
521	002544	001401				BEQ	WCCON		:YES
522	002546	000431				BR	OPDAR		:CONT

523	002550										
524	002550	104402	002554			WCCON:	TYPE	..+2		:.ASCIZ <'5><12>'WD CT ''	
525	002566	104400					RDOCT				
526	002570	012637	001174				MOV	(6)+,NUMS		:GET NUMBER	
527	002574	005737	001174				TST	NUMS		:IS IT 0?	
528	002600	001763					BEQ	WCCON		:YES ASK AGAIN FOR LENGTH	
529	002602	013702	001220				MOV	SAVWC,R2		:GET STANDARD WC FOR -A- PORT	
530	002606	005202				1\$:	INC	R2			
531	002610	020237	001174				CMP	R2,NUMS		:IS NUMS LESS THAN SWRDC	
532	002614	101755					BLOS	WCCON		:YES ASK FOR COUNT AGAIN	
533	002616	013737	001174	001150			MOV	NUMS,SWRDC		:OPERATING WORD COUNT	
534	002624	013737	001150	001134			MOV	SWRDC,WRDC			
535											
536	002632					OPDAR:	TYPE	..+2		:.ASCIZ <'5><12>'OPTIONAL DSK ADDR''	
537	002632	104402	002636				JSR	PC,CMPY		:COMPARE FOR YES	
538	002662	004737	003234				BNE	OPPAT		:ANS IS NO	
539	002666	001034					BIS	#BIT5,FLAG		:SET OPTIONAL DMA FLAG	
540	002670	052737	000040	001132			TYPE	..+2		:.ASCIZ <'5><12>'DSK ADDR''	
541	002676	104402	002702				RDOCT				
542	002716	104420					MOV	(6)+,NUMS		:GET NUMBER	
543	002720	012637	001174				TST	RS04DT		:IS THIS A RS04?	
544	002724	005737	001172				BEQ	1\$		:NO	
545	002730	001404					CMP	#17777,NUMS		:IS ADD. CORRECT?	
546	002732	022737	017777	001174			BR	2\$		:GET OUT	
547	002740	000403				1\$:	CMP	#7777,NUMS		:IS ADD. CORRECT?	
548	002742	022737	007777	001174		2\$:	BLOS	OPDAR		:NO	
549	002750	101730					MOV	NUMS,TDMA		:TEMP SECTOR REGISTER	
550	002752	013737	001174	001146							

551	002760				OPPAT:				
552	002760	104402	002764			TYPE	..+2		;.ASCIZ <15><12>'DATA PATTERN # ''
553	003006	104420				RDOCT			
554	003010	012637	001174			MOV	(6)+,NUMS		;.GET NUMBER
555	003014	022737	000023	001174		CMP	#23,NUMS		;.TEST FOR CORRECT NO
556	003022	101756				BLOS	OPPAT		;.ASK AGAIN
557	003024	005037	001142			CLR	PATNU		;.CLEAR PATTERN #
558	003030	022737	000022	001174		CMP	#22,NUMS		
559	003036	001411				BEQ	OPWRT		;.DATA PATTERN UNDER PROGRAM CONTROL
560	003040	052737	100000	001132		BIS	#BIT15,FLAG		;.SET PROGRAM FLAG
561	003046	013737	001174	001142		MOV	NUMS,PATNU		;.OPERATOR WANTS TO SELECT DATA
562	003054	000241				CLC			
563	003056	006137	001142			ROL	PATNU		
564									
565	003062				OPWRT:				
566	003062	104402	003066			TYPE	..+2		;.ASCIZ <15><12>'WRITE''
567	003076	004737	003234			JSR	PC,CMPLY		;.COMPARE FOR YES
568	003102	001003				BNE	OPRD		;.ASK ABOUT WRITE CHECK
569	003104	052737	040000	001132		BIS	#BIT14,FLAG		;.YES SET FLAG BIT
570									
571	003112				OPRD:				
572	003112	104402	003116			TYPE	..+2		;.ASCIZ <15><12>'READ''
573	003126	004737	003234			JSR	PC,CMPLY		;.COMPARE FOR YES
574	003132	001003				BNE	OPWCK		
575	003134	052737	010000	001132		BIS	#BIT12,FLAG		;.SET FLAG TO READ
576									
577	003142				OPWCK:				
578	003142	104402	003146			TYPE	..+2		;.ASCIZ <15><12>'WRITE CK''
579	003162	004737	003234			JSR	PC,CMPLY		;.COMPARE FOR YES
580	003166	001003				BNE	ADTST		
581	003170	052737	020000	001132		BIS	#BIT13,FLAG		
582									
583	003176	032737	004000	001132	ADTST:	BIT	#BIT11,FLAG		;.ARE WE IN MULTI DRIVE MODE
584	003204	001402				BEQ	EXMFLG		;.BRANCH IF NO.
585	003206	004737	001620			JSR	PC,DRVENO		;.GET DRIVES TO BE TESTED
586	003212	042737	000004	001132	EXMFLG:	BIC	#BIT2,FLAG		;.CLEAR XFER MODE FLAG
587	003220	032737	002000	001132	1\$:	BIT	#BIT10,FLAG		;.TEST FOR DATA TEST ONLY
588	003226	001414				BEQ	ADT1		;.DO COMPLETE TEST
589	003230	000137	003660			JMP	DATAT		;.DO DATA TEST ONLY
590									
591	003234	104402	000763		CMPLY:	TYPE	,YORN		
592	003240	104422				RDLIN			
593	003242	122737	000131	017246		CMFB	#'Y,INPUT		;.TEST FOR YES
594	003250	000207				RTS	PC		
595	003252	052737	100000	001222	ADTL:	BIS	#BIT15,FLAG3		;.SET LOOP IN ADDRESS TEST FLAG GOT HERE
596									;.BECAUSE PROGRAM WAS STARTED AT 260

```

597 :RH70 ADDRESS TEST #1 (TRACK AND SECTOR SELECTION TEST)
598 :WRITE 100(OCTAL) RS03, 200(OCTAL) RS04, WORDS IN EACH SECTOR
599 :THE WORD CONTAINS THE ADDRESS OF EACH SECTOR
600 :WHEN THE COMPLETE DISK IS WRITTEN READ
601 :BACK EACH SECTOR AND COMPARE FOR THE CORRECT
602 :DATA IN THE SECTOR
603 :PS IS AT LEVEL 7 SO NO INTERRUPTS
604
605 003260
606
607
608
609 003260 104400
610 003262 012737 000340 177776
611 003270 005037 001024
612 003274 012737 020000 017326
613 003302 052737 000400 001132
614 003310 013737 001150 001154
615 003316 005037 001140
616 003322 104426
617 003324 012737 000200 001134
618 003332 012737 000200 001150
619 003340 005737 001172
620 003344 001006
621 003346 012737 000100 001134
622 003354 012737 000100 001150
623 003362 013737 017326 001144
624 003370 104414
625 003372 013700 017326
626 003376 013701 001134
627 003402 013720 001140
628 003406 005301
629 003410 001374
630 003412 012737 000061 001176
631 003420 104416
632 003422 105777 175404
633 003426 100375
634 003430 005777 175376
635 003434 100010
636 003436 012737 003370 001010
637 003444 052737 001000 001132
638 003452 104430
639 003454 104034
640 003456 104400
641 003460 004737 007204
642 003464 000741
643 003466 104400
  
```

```

ADT1: ;ADDRESS TEST
:*****
:TEST 1 ADDRESS TEST
:*****
TST1: SCOPE
ADT1A: MOV #340,PS ;LOCK UP PS
CLR TEMP2
MOV #20000,OUTBUF ;START BUF AT 20000
BIS #BIT8,FLAG ;SET TEST FLAG
MOV SWRDCT,SAVE ;SAVE STD WD COUNT
CLR DMA ;CLEAR DISK ADD
CLRDV ;INIT DRIVE
MOV #200,WRDCT ;SETUP WC
MOV #200,SWRDCT
TST RS04DT ;IS THIS A RS04?
BNE Z$ ;YES
1$: MOV #100,WRDCT ;SETUP WORD COUNT
MOV #100,SWRDCT
2$: MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
SEABUF: ERCLR ;CLEAR RS REGISTERS IF ERROR
MOV OUTBUF,R0 ;SET UP ADDRESS BUFFER
MOV WRDCT,R1
XSEABUF: MOV DMA,(0)+ ;LOAD OUTBUF WITH DATA TO BE WRITTEN
DEC R1 ;FILL OUTBUF
BNE XSEABUF ;WITH DATA
MOV #61,CMD ;WRITE NO I/E
DKCMD ;GO WRITE
TSTB @RSCS1 ;CHECK FOR READY
BPL -4
TST @RSCS1 ;TEST FOR ERROR
BPL WRNEXB ;BRANCH IF NO ERROR
MOV #SEABUF,LAD ;SET UP LOOP ADDRESS
BIS #BIT9,FLAG ;SET ERROR BIT IN FLAG
LOGW ;LOG WRITE ERROR
HLT !WC!DA!BA
WRNEXB: SCOPE
JSR PC,DISBUF ;SET UP NEXT DISK ADDR.
BR SEABUF ;WRITE NEXT SECTOR
RRDSEC: SCOPE
  
```

```

644 003470 104414 RDSECT: ERCLR ;CLEAR ERRORS
645 003472 012737 000071 001176 MOV #71,CMD ;READ NO I/E
646 003500 104416 DKCMD ;DO A READ
647 003502 105777 175324 TSTB @RSCS1 ;CHECK FOR READY
648 003506 100375 BPL -4 ;NOT READY BRANCH BACK
649 003510 005777 175316 TST @RSCS1 ;TEST FOR ERROR
650 003514 100006 BPL ADHGT ;BRANCH IF NO ERROR
651 003516 052737 001000 001132 BIS #BIT9,FLAG ;SET ERROR FLAG
652 003524 104432 LOGR ;LOG READ ERROR
653 003526 104014 HLT !WC!DA
654 003530 104400 SCOPE
655 003532 013702 017326 ADHGT: MOV OUTBUF,R2
656 003536 005737 001172 TST RS04DT ;RS04?
657 003542 001403 BEQ 1$ ;NO
658 003544 012703 000200 MOV #200,R3 ;YES
659 003550 000402 BR SANHT ;CONT
660 003552 012703 000100 $: MOV #100,R3
661 003556 023712 001140 SANHT: CMP DMA,(2) ;CMP FOR CORRECT ADDR.
662 003562 001004 BNE ADERR ;BRANCH IF DATA DID NOT COMPARE
663 003564 005722 TST (2)+ ;GET NEXT ADDRESS OF INBUF
664 003566 005303 DEC R3 ;DEC SECTOR COUNT
665 003570 001372 BNE SANHT ;TEST NEXT WORD
666 003572 000412 BR CHKADT
667 003574 013701 001140 ADERR: MOV DMA,GOOD ;CORRECT ADDRESS
668 003600 011200 MOV (2),BAD ;DATA IN ERROR
669 003602 104000 HLT ;DISK ADD DID NOT MATCH WRITTEN ADDRESS
670 003604 104436 LOGC ;LOG COMPARE ERROR
671 003606 004737 014112 JSR PC,PRNT ;INHIBIT TYPEOUT?
672 003612 001002 BNE CHKADT ;YES
673 003614 104402 000757 TYPE ,CRLF
674
675 ;*****REPORT ONLY ONE ERROR PER SECTOR*****
676
677 003620 104400 CHKADT: SCOPE
678 003622 004737 007204 JSR PC,DISBUF ;SET UP NEXT DISK BUFFER
679 003626 000717 BR RRDSEC ;CHECK NEXT SECTOR
680 003630 013737 001154 001150 MOV SAVE,SWRDCT ;GET STD WD COUNT
681 003636 042737 000400 001132 BIC #BIT8,FLAG ;CLEAR TEST FLAG
682 003644 032737 100000 001222 BIT #BIT15,FLAG3 ;DOES OPERATOR WANT TO LOOP ON TEST
683 003652 001402 BEQ +6 ;NO
684 003654 000137 003262 JMP ADT1A ;YES
685 003660 DATAT: ;DATA TEST
686
687 ;*****
688 ;TEST 2 DATA TEST
689 ;*****
689 003660 104400 TST2: SCOPE
690 ;XYZ*****?*****
691 ;*****
692 ;*****
693 ;XYZ*****?*****
694 ;*****
695 ;*****
696 ;XYZ*****?*****
697 ;*****
698 003662 012737 177777 001024 MOV #-1,TEMP2 ;TEST 2 INDICATOR
699 003670 013737 001074 001026 MOV STAMEM,TEMP3
    
```

700	003676	013737	001150	001134	LDAT3:	MOV	SWRDCT,WRDCT	
701	003704	005037	001140		LDAT2:	CLR	DMA	;CLEAR DISK ADDRESS
702	003710	104426				CLR DV		;CLEAR RS REGISTERS
703	003712	004737	011376			JSR	PC,APORT	
704	003716	004737	007130		2\$:	JSR	PC,VECTRR	;SETUP INT VECTOR
705	003722	012777	000340	175140		MOV	#340,@STATUS	;SET DISK STATUS REG LOC (206)
706	003730	012737	003762	001156		MOV	#LDAT,HRDR	;SETUP FOR HARD ERROR RETURN
707	003736	013737	017326	001144		MOV	OUTBUF,BUF	;SETUP OUTPUT BUFFER
708	003744	052737	000003	001132		BIS	#3,FLAG	;SET COUNTER TO 3
709	003752	004537	007650			JSR	RS,PASEL	;SET UP DATA BUFFERS
710	003756	005037	001204		LDAT1:	CLR	LOPCNT	;CLEAR ERROR FLAG
711	003762	104414			LDAT:	ERCLR		;CLEAR RS REG. IF ERROR
712	003764	004537	006546			JSR	RS,OPDSEL	;SET UP DISK ADDRESS
713	003770	032737	040000	001132		BIT	#BIT14,FLAG	;TEST FOR WRITE
714	003776	001462				BEQ	SLH	;TEST FOR READ
715	004000	012737	000161	001176		MOV	#161,CMD	;WRITE WITH I/E
716	004006	104416				DKCMD		;DO A WRITE
717	004010	004737	011600			JSR	PC,WATT	;WAIT FOR INTERRUPT
718	004014	012737	003762	001010		MOV	#LDAT,LAD	;SETUP SCOPE LOOP
719	004022	032737	001000	001132		BIT	#BIT9,FLAG	;WAS THERE AN ERROR?
720	004030	001427				BEQ	WRXBL	;CONT
721	004032	104430				LOGW		;LOG WRITE ERROR
722	004034	005237	001204			INC	LOPCNT	;SET ERROR FLAG
723	004040	022737	000001	001204		CMP	#1,LOPCNT	;IS THIS THE FIRST TIME?
724	004046	001000				BNE	2\$	;NO
725	004050	004737	014112		2\$:	JSR	PC,PRNT	;TYPE ?
726	004054	001004				BNE	1\$	;NO
727	004056	104402	000620			TYPE	.DATA	
728	004062	104402	000630			TYPE	.WRERR	
729	004066	104444			1\$:	HLT	!DS!DA	;WRITE ERROR
730	004070	005337	001132			DEC	FLAG	;DEC COUNTER
731	004074	032737	000003	001132		BIT	#3,FLAG	;DONE YET WITH 3RD TRY?
732	004102	001327				BNE	LDAT	;NOT 3 TRIES YET? TRY AGAIN
733	004104	004737	011444			JSR	PC,WTNO	;TYPE CAN NOT WRITE

734	004110	005737	001204		WRXBL:	TST	LOPCNT		:WAS THERE AN ERROR?
735	004114	001402				BEQ	WRX1		:NO
736	004116	004737	011622			JSR	PC,TYPREC		:TYPE RECOVERED
737	004122	005037	001204		WRX1:	CLR	LOPCNT		:CLEAR ERROR FLAG
738	004126	104400				SCOPE			
739	004130	052737	000003	001132		BIS	#3,FLAG		:CLEAR RETRY COUNT
740	004136	004737	007204			JSR	PC,DISBUF		:SET BUFFER FOR WRITE CHECK
741	004142	000705				BR	LDT1		
742	004144	104400			SLH:	SCOPE			
743	004146	104414			SLH2:	ERCLR			:CLEAR RS REG IF ERRORS
744	004150	004537	006546			JSR	R5,OPDSEL		:IS THE OPERATOR SELECTING THE TRACK
745	004154	032737	020000	001132		BIT	#BIT13,FLAG		:TEST FOR WRITE CHECK
746	004162	001002				BNE	1\$		:YES
747	004164	000137	004470			JMP	ESH1		:NO
748	004170	013737	017326	001144	1\$:	MOV	OUTBUF,BUF		:SET UP CURRENT ADDRESS
749	004176	012737	000151	001176		MOV	#151,CMD		:WRITE CHECKWITH I/E
750	004204	104416				DKCMD			:GO WRITE CHECK
751	004206	004737	011600			JSR	PC,WATT		:WAIT FOR INTERRUPT
752	004212	032737	001000	001132	XESH:	BIT	#BIT9,FLAG		:IS THERE AN ERROR?
753	004220	001505				BEQ	1\$		:NO ERROR
754	004222	005737	001204			TST	LOPCNT		:1ST ERROR?
755	004226	001001				BNE	2\$		:NO
756	004230	104434				LOGWC			:YES LOG ERROR
757	004232	032737	000100	177570	2\$:	BIT	#BIT6,SWR		:TYPE ALL ERRORS?
758	004240	001007				BNE	3\$		:YES
759	004242	032737	001000	177570		BIT	#BIT9,SWR		:LOOP ON ERROR?
760	004250	001003				BNE	3\$		:YES
761	004252	005737	001204			TST	LOPCNT		:FIRST ERROR?
762	004256	001056				BNE	10\$		:NO
763	004260	004737	014112		3\$:	JSR	PC,PRNT		:TYPE OUT?
764	004264	001052				BNE	4\$		:NO
765	004266	104402	000620			TYPE	.DATA		
766	004272	104402	000642			TYPE	.WCKERR		
767	004276	017702	174536			MOV	@RSBA,R2		:GET CORRECT BA
768	004302	023702	017326			CMP	OUTBUF,R2		:DID A WD GET XFERED?
769	004306	001406				BEQ	9\$		:NO
770	004310	032737	000400	177570		BIT	#BIT8,SWR		:XFER MODE?
771	004316	001002				BNE	9\$		:YES
772	004320	162702	000002			SUB	#2,R2		
773	004324	004737	014112		9\$:	JSR	PC,PRNT		:TYPEOUT ERRORS?
774	004330	001030				BNE	4\$		:NO
775	004332	005737	001216			TST	MMAVA		:IS MEM MGMT AVAILABLE?
776	004336	001402				BEQ	7\$		:NO



777	004340	004737	006762			JSR	PC,PHYCOV	:YES GET VITURAL ADDR
778	004344	010237	001226	7\$:		MOV	R2,WORK	:GET BA
779	004350			8\$:				
780	004350	104402	004354			TYPE	..+2	:.ASCIZ <15><12>'(BA)-'
781	004364			6\$:				
782	004364	017746	174636			MOV	@WORK,-(6)	:PUT @WORK ON STACK
783	004370	104404				TYPE0		:TYPE STACK IN OCTAL
784	004372	104402	004376			TYPE	..+2	:.ASCIZ 'WC='
785	004404	017746	174426			MOV	@RSWC,-(6)	:PUT @RSWC ON STACK
786	004410	104404				TYPE0		:TYPE STACK IN OCTAL
787	004412	104026		4\$:		HLT	!DA!DB!BA	:NOTE: BA REG. = +2 OF ACTUAL MEMORY
788								:LOC AFTER WORDS HAVE BEEN XFERED
789	004414	005237	001204	10\$:		INC	LOPCNT	:INC ERROR COUNT
790	004420	022737	000010	001204		CMP	#10,LOPCNT	: '0 TRYS YET?
791	004426	001247				BNE	SLH2	: '0
792	004430	004737	006720			JSR	PC,NOREC	:TYPE UNRECOVERABLE
793	004434	005737	001204	\$:		TST	:OPCNT	:ANY ERRORS?
794	004440	001402				BEQ	5\$	:NO
795	004442	004737	011622			JSR	PC,TYPREC	:TYPE RECOVERED
796	004446	005037	001204	5\$:		CLR	LOPCNT	:CLEAR ERROR COUNTER
797	004452	104400				SCOPE		
798	004454	012737	004146	001010		MOV	#SLH2,LAD	:SETUP LOOP ADDRESS
799	004462	004737	007204			JSR	PC,DISBUF	:SET UP THE DISK BUFFER
800	004466	000422				BR	SLH2A	
801	004470	004537	011312	ESH1:		JSR	R5,CLEAR	:CLEAR BUFFER
802	004474	004537	006546	ESH:		JSR	R5,OPDSEL	:OPERATOR SELECTED DISK ADDRESS?
803	004500	032737	010000	001132		BIT	#BIT12,FLAG	:TEST FOR READ
804	004506	001002				BNE	1\$	:YES
805	004510	000137	004724			JMP	MSTR	:NO READ
806	004514	104400		1\$:		SCOPE		
807	004516	042737	000003	001132		BIC	#3,FLAG	:CLEAR RE-READ COUNT
808	004524	005037	001204			CLR	LOPCNT	:CLEAR FLAG
809	004530	000137	004540			JMP	DSKRD	:CONT
810	004534	000137	004146	SLH2A:		JMP	SLH2	
811	004540	104414		DSKRD:		ERCLR		:CLEAR RS REG IF ERRORS
812	004542	012737	000171	001176		MOV	#171,CMD	:READ WITH I/E
813	004550	104416				DKCMD		:READ
814	004552	004737	011600			JSR	PC,WATT	
815	004556	032737	010000	177570		BIT	#10000,SWR	:COMPARE?
816	004564	001007				BNE	TAG	:NO
817	004566	032737	000004	001132		BIT	#BIT2,FLAG	:COMPARE?
818	004574	001003				BNE	TAG	:NO
819	004576	004537	010414			JSR	R5,COMPARE	:COMPARE
820	004602	000400				BR	FLH	

821	004604					TAG:			
822	004604	032737	001000	001132		ELH:	BIT	#BIT9,FLAG	:IS THERE AN ERROR?
823	004612	001433					BEQ	ADDR	:NO
824	004614	032737	000100	177570			BIT	#BIT6,SWR	:SOFT ERROR TYPEOUT?
825	004622	001003					BNE	3\$	:YES
826	004624	005737	001204				TST	LOPCNT	:FIRST ERROR?
827	004630	001011					BNE	2\$	:NO
828	004632	004737	014112			3\$:	JSR	PC,PRNT	:TYPEOUT?
829	004536	001004					BNE	1\$	:NO
830	004640	104402	000620				TYPE	.DATA	
831	004644	104402	000657				TYPE	.RDERR	
832	004650	104432				1\$:	LOGR		:LOG READ ERROR
833	004652	104044					HLT	.DS!DA	
834	004654	104400				2\$:	SCOPE		
835	004656	005237	001204				INC	LOPCNT	:COUNT ERRORS
836	004662	022737	000010	001204			CMP	#10,LOPCNT	:LAST RETRY?
837	004670	001323					BNE	DSKRD	:NO
838	004672	004737	006720				JSR	PC,NOREC	:TYPE UNRECOVERABLE
839	004676	000137	004716			4\$:	JMP	ADR1	:CONT
840	004702	104400				ADDR:	SCOPE		
841	004704	005737	001204				TST	LOPCNT	:ANY ERRORS?
842	004710	001402					BEQ	ADR1	:NO
843	004712	004737	011622				JSR	PC,TYPREC	:TYPE RECOVERED
844	004716	004737	007204			ADR1:	JSR	PC,DISBUF	:GO SET UP DISK BUFFER.
845	004722	000500					BR	READCT	:CONT. READING
846	004724	005737	001132			MSTR:	TST	FLAG	
847	004730	100467					BMI	3\$	:OPERATOR SELECTED PATTERN
848									
849									
850									
851									
852									
853									
854									
855									
856									
857	004732	013746	001134				MOV	WRDCT,-(SP)	
858	004736	000316					SWAB	(SP)	
859	004740	042716	177400				BIC	#177400,(SP)	
860	004744	006216					ASR	(SP)	
861	004746	006216					ASR	(SP)	
862	004750	006216					ASR	(SP)	
863	004752	006216					ASR	(SP)	
864	004754	061637	001026				ADD	(SP),TEMP3	
865	004760	023737	001022	001026			CMP	TEMP1,TEMP3	
866	004766	003424					BLE	5\$	
867	004770	061637	001026				ADD	(SP),TEMP3	
868	004774	023737	001022	001026			CMP	TEMP1,TEMP3	
869	005002	003416					BLE	4\$	
870	005004	162637	001026				SUB	(SP)+,TEMP3	
871	005010	005037	017326				CLR	OUTBUF	
872	005014	013746	001026				MOV	TEMP3,-(SP)	
873	005020	062737	020000	017326	6\$:		ADD	#20000,OUTBUF	
874	005026	005316					DEC	(SP)	
875	005030	001373					BNE	6\$	
876	005032	005726					TS*	(SP)+	

877	005034	000137	003676			JMP	LDAT3	
878	005040				4\$:			
879	005040	005726			5\$:	TST	(SP)+	
880	005042	062737	000002	001142		ADD	#2,PATNU	:INC PATTERN INDEX
881	005050	022737	000044	001142		CMP	#44,PATNU	
882	005056	001402				BEQ	.+6	
883	005060	000137	003660			JMP	DATAT	:NOT LAST PATTERN EXIT
884	005064	005037	001142			CLR	PATNU	:LAST PATTERN EXIT
885	005070	032737	000002	177570		BIT	#BIT11,SWR	:DATA TEST ONLY?
886	005076	001006				BNE	2\$	:YES
887	005100	032737	002000	001132		BIT	#BIT10,FLAG	:DATA TEST ONLY?
888	005106	001404				BEQ	1\$	:NO
889	005110	005137	001120		3\$:	COM	AQB1	:ALTERNATE PORTS
890	005114	000137	006154		2\$:	JMP	EXTPPR	:LOOP
891	005120	000137	005136		1\$:	JMP	RANEX	:DO NEXT TEST
892								
893	005124	000137	004474		HEADCT:	JMP	FSH	:CONT. READING
894	005130	052737	100000	001222	RANEL:	BIS	#BIT15,FLAG3	:SET LOOP IN RANDOM TEST GOT
895								:HERE BY STARTING AT LOC 264

```

896 005136 RANEX: :RANDOM ADDRESS DATA TEST
897 :THIS PROGRAM WRITES, WRITECHECKS AND READS 1 SECTOR OF RANDOM DATA FROM RANDOM DISK
898 :ADDRESSES. THIS TEST WILL MAKE 1000(10) PASSES BEFORE IT IS COMPLETED
899
900 :*****
901 :TEST 3 RANDOM ADDRESS RANDOM DATA TEST
902 :*****
903 005136 104400 TST3: SCOPE
904 :XYZ*****?*****
905 :*****
906 :*****
907 :XYZ*****?*****
908 :*****
909 :*****
910 :XYZ*****?*****
911 :*****
912 005140 005037 001024 CLR TEMP2 :NOMORE A TEST 2
913 005144 052737 000400 001132 BIS #BIT8,FLAG :SET TEST FLAG
914 005152 012737 020000 017326 MOV #20000,OUTBUF :GET STARTING ADDR OF BUF
915 005160 013737 017326 001122 MOV OUTBUF,VADDR :SAVE BUFFER ADDR
916 005166 005737 001216 TST MMAVA :MEM MGMT AVAILABLE?
917 005172 001402 BEQ 1$ :NO
918 005174 005037 177572 CLR @MSRO :TURN IT OFF
919 005200 012737 000042 001142 1$: MOV #42,PATNU :DO RANDOM COMPARE
920 005206 104426 CLRDV :INIT DRIVE
921 005210 012737 176030 001162 MOV #-1000.,PASSC :SET UP PASS COUNT
922 005216 012737 005774 001156 MOV #WRERR,WRDR :SET UP FOR HARD ERROR
923 005224 004737 007130 JSR PC,VECTRR :SETUP INTERRUPT VECTOR
924 005230 012777 000340 173632 MOV #340,@STATUS
925 005236 012737 005356 001010 WRLG1: MOV #WRERR,LAD :SETUP LOOP ADDRESS
926 005244 012737 000001 001226 MOV #1,WORK :SET UP RANDOM GENERATOR WORD
927 005252 013701 017326 MOV OUTBUF,R1
928 005256 004537 010010 JSR R5,RANDOM :GENERATE RANDOM DATA
929 005262 017737 012040 001140 MOV @OUTBUF,DMA :SET UP DISK ADDRESS
930 005270 042737 170000 001140 BIC #170000,DMA
931 005276 052737 000003 001132 BIS #3,FLAG :SET COUNTER
932 005304 012737 000200 001134 MOV #200,WRDCT :RS04
933 005312 005737 001172 TST RS04DT :RS04?
934 005316 001003 BNE 2$ :YES
935 005320 012737 000100 001134 MOV #100,WRDCT :SET UP WORD COUNT =1SECTOP
936 005326 013737 001134 001226 2$: MOV WRDCT,WORK :GENERATE RANDOM BUFFER
937 005334 013701 017326 MOV OUTBUF,R1
938 005340 004537 010010 JSR R5,RANDOM
939 005344 013737 017326 001144 MOV OUTBUF,BUF :SET UP OUTPUT BUFFER
940 005352 005037 001204 CLR LOPCNT :CLR ERROR FLAG
    
```

941	005356	104414			WRERR:	ERCLR		
942	005360	012737	000161	001176		MOV	#161,CMD	:WRITE WITH I/E
943	005366	104416				DKCMD		:WRITE
944	005370	004737	011600			JSR	PC,WAIT	:WAIT FOR INTERRUPT
945	005374	032737	001000	001132	2\$:	BIT	#BIT9,FLAG	:WAS THERE AN ERROR?
946	005402	001435				BEQ	WRCK1	:NO
947	005404	032737	000100	177570		BIT	#BIT6,SWR	:TYPE RETRY ?
948	005412	001004				BNE	5\$	:YES
949	005414	005737	001204			TST	LOPCNT	:FIRST TIME?
950	005420	001013				BNE	6\$	:NO
951	005422	104430				LOGW		:LOG WRITE ERROR
952	005424	005237	001204		5\$:	INC	LOPCNT	:SET ERROR FLAG
953	005430	004737	014112			JSR	PC,PRNT	:TIMEOUT?
954	005434	001004				BNE	3\$	:YES
955	005436	104402	000670			TYPE	,RANDM	
956	005442	104402	000630			TYPE	,WRERR	
957	005446	104044			3\$:	HLT	!DS!DA	
958	005450	104400			6\$:	SCOPE		
959	005452	005337	001132			DEC	FLAG	
960	005456	032737	000003	001132		BIT	#3,FLAG	
961	005464	001334				BNE	WRERR	:RETRY
962	005466	004737	011444			JSR	PC,WTNO	:TYPE CAN NOT WRITE
963	005472	000137	006076			JMP	EXRAX	:GET NEW NUMBER

964	005476	005737	001204		WRRCK1:	TST	LOPCNT		; ANY ERRORS?
965	005502	001402				BEQ	1\$		; NO
966	005504	004737	011622			JSR	PC, TYPREC		; TYPE RECOVERED
967	005510	104400			1\$:	SCOPE			
968	005512	005037	001204			CLR	LOPCNT		; CLEAR LOOP COUNT
969	005516	104414			WRRCK:	ERCLR			; CLEAR RS REG IF ERRORS
970	005520	012737	000151	001176		MOV	#151, CMD		; WRITE CHECK WITH I/E
971	005526	104416				DKCMD			; WRITE CHECK
972	005530	004737	011600			JSR	PC, WATT		; WAIT FOR INTERRUPT
973	005534	032737	001000	001132	4\$:	BIT	#BIT9, FLAG		; ERROR?
974	005542	001453				BEQ	1\$		; NO
975	005544	032737	000100	177570		BIT	#BIT6, SWR		; TYPE ALL RETRYS?
976	005552	001003				BNE	2\$		; YES
977	005554	005737	001204			TST	LOPCNT		; FIRST ERROR?
978	005560	001030				BNE	5\$		; NO
979	005562	104434			2\$:	LOGWC			; LOG WRITE CK
980	005564	004737	014112			JSR	PC, PRNT		; TYPEOUT?
981	005570	001052				BNE	6\$		; NO
982	005572	104402	000670			TYPE	, RANDM		
983	005576	104402	000642			TYPE	, WCKERR		
984	005602	017737	173232	001226		MOV	@RSBA, WORK		; GET CORRECT BA
985	005610	162737	000002	001226		SUB	#2, WORK		
986	005616	104402	005622			TYPE	, +2		; .ASCIZ <15><12>'(BA)='
987	005632	017746	173370			MOV	@WORK, -(6)		; PUT @WORK ON STACK
988	005636	104404				TYPEO			; TYPE STACK IN OCTAL
989	005640	104026				HLT	!DB!DA!BA		; BA=MEMORY LOC +2 OF ACTUAL WORD
990	005642	005237	001204		5\$:	INC	LOPCNT		; INC RETRY COUNT
991	005646	022737	000010	001204		CMP	#10, LOPCNT		; LAST ONE YET?
992	005654	001320				BNE	WRRCK		; NO
993	005656	104402	006651			TYPE	, UNRECO		
994	005662	005037	001204			CLR	LOPCNT		; CLEAR LOPCNT
995	005666	000137	006076			JMP	EXRAX		; GET NEW NUMBER
996	005672	005737	001204		1\$:	TST	LOPCNT		; ANY ERRORS?
997	005676	001407				BEQ	6\$		; NO
998	005700	104402	000702			TYPE	, RECOV		
999	005704	013746	001204			MOV	LOPCNT, -(6)		; GET NUMBER
1000	005710	104406				TYPES			; TYPE IT
1001	005712	104402	000757			TYPE	, CRLF		
1002	005716	104400			6\$:	SCOPE			
1003	005720	052737	000003	001132		BIS	#3, FLAG		; SET COUNTER

1004	005726	104400				SCOPE		
1005	005730	005037	001204			CLR	LOPCNT	:CLEAR COUNTER
1006	005734	004537	011312			JSR	RS,CLEAR	:CLEAR BUFFER
1007	005740	104414			RREAD:	ERCLR		:CLEAR RS REG IF ERRORS
1008	005742	012737	000171	001176		MOV	#171,CMD	:READ WITH I/E
1009	005750	104416				DKCMD		:READ
1010	005752	004737	011600			JSR	PC, WATT	
1011	005756	032737	010000	177570		BIT	#BIT12,SWR	:COMPARE ?
1012	005764	001003				BNE	TAG1	:NO
1013	005766	004537	010414			JSR	RS,COMPARE	:YES
1014	005772	000400				BR	RWRED	:CONT
1015	005774				TAG1:			
1016	005774	032737	001000	001132	RWRED:	BIT	#BIT9,FLAG	:IS THERE AN ERROR?
1017	006002	001435				BEQ	EXRAX	:NO
1018	006004	104432				LOGR		:LOG READ ERR
1019	006006	032737	000100	177570	1\$:	BIT	#BIT6,SWR	:TYPE ALL ERRORS?
1020	006014	001016				BNE	2\$	:YES
1021	006016	032737	001000	177570		BIT	#BIT9,SWR	:LOOP ON ERROR?
1022	006024	001012				BNE	2\$	:YES
1023	006026	005737	001204			TST	LOPCNT	:FIRST ERROR?
1024	006032	001010				BNE	3\$	:NO
1025	006034	004737	014112			JSR	PC,PRNT	:TYPEOUT?
1026	006040	001004				BNE	2\$	:NO
1027	006042	104402	000670			TYPE	,RANDM	
1028	006046	104402	000657			TYPE	,RDERR	
1029	006052	104006			2\$:	HLT	!DB!DA	
1030	006054	104400			3\$:	SCOPE		
1031	006056	005237	001204			INC	LOPCNT	:UPDATE COUNTER
1032	006062	022737	000010	001204		CMP	#10,LOPCNT	:LAST TRY YET?
1033	006070	001323				BNE	RREAD	:RETRY
1034	006072	004737	006720			JSR	PC,NOREC	:TYPE UNRECOVERABLE
1035	006076	005737	001204		EXRAX:	TST	LOPCNT	:ANY ERRORS?
1036	006102	001402				BEQ	EXRXX	:NO
1037	006104	004737	011622			JSR	PC,TYPREC	:TYPE RECOVERED
1038	006110	104400			EXRXX:	SCOPE		
1039	006112	005237	001162			INC	PASSC	:+1 PASS COUNT
1040	006116	001402				BEQ	1\$	:IS TEST DONE?
1041	006120	000137	005236			JMP	WRLG1	:NO
1042	006124	005037	001142		1\$:	CLR	PATNU	:END OF TEST
1043	006130	042737	000400	001132		BIC	#BIT8,FLAG	:CLEAR TEST FLAG
1044	006136	032737	100000	001222		BIT	#BIT15,FLAG3	:LOOP ON THIS TEST?
1045	006144	001402				BEQ	.+6	:NO
1046	006146	000137	005136			JMP	RANEX	:YES

```

1047 :CHECK FOR MULTI DISK MODE
1048 :IF IN MULTI DISK MODE REPORT 'END'
1049 :IF LAST DISK ON SYSTEM HAS BEEN EXERCISED.
1050
1051 :*****
1052 :TEST 4 TST FOR MULTI DISK MODE
1053 :*****
1054 006152 104400 TST4: SCOPE
1055 006154 005037 001140 EXTPPR: CLR DMA
1056 006160 005037 001024 CLR TEMP2
1057 006164 104426 CLRDV ;INIT DRIVE
1058 006166 032737 004000 001132 BIT #BIT11,FLAG ;ARE WE IN MULTI DISK MODE
1059 006174 001404 BEQ EXTPP ;NO REPORT 'END'
1060 006176 004737 002130 JSR PC,TRYNX ;YES TEST FOR ALL DRIVES
1061 006202 000137 003212 JMP EXMFLG ;RESTART TESTING OF DRIVES
1062 006206 004737 002204 EXTPP: JSR PC,DONEE ;GET PASS COUNT
1063
1064 :THIS ROUTINE CLEARS THE DRIVE
1065 :REGISTERS IF THERE WAS AN ERROR
1066
1067 006212 032737 001000 001132 .ERCLR: BIT #BIT9,FLAG ;ANY ERRORS?
1068 006220 001404 BEQ 1$ ;NO
1069 006222 104426 CLRDV ;CLEAR ALL ERRORS
1070 006224 042737 001000 001132 BIC #BIT9,FLAG ;CLEAR ERROR FLAG
1071 006232 000002 1$: RTI ;EXIT
1072
1073 :ENTER DISK HANDLER BY THE TRAP INSTRUCTION
1074 :ARGUMENT TO TRAP INSTRUCTION IS TWO ORDER
1075 :BYTE OF THE CONTROL REGISTER.
1076
1077 006234 013777 001140 172600 .DKCMD: MOV DMA,@RSDA ;LOAD DISK ADD
1078 :*****
1079 :*****
1080 :*****
1081 :*****
1082 :*****
1083 :*****
1084 :*****
1085 :*****
1086 006242 005737 001024 TST TEMP2
1087 006246 001422 BEQ 5$
1088 006250 013746 001026 MOV TEMP3,-(6)
1089 006254 005077 172602 CLR @RSBAE
1090 006260 005037 001030 CLR TEMP4
1091 006264 062737 020000 001030 6$: ADD #20000,TEMP4
1092 006272 013737 001030 001144 MOV TEMP4,BUF
1093 006300 103002 BCC 7$
1094 006302 005277 172554 INC @RSBAE
1095 006306 005316 7$: DEC (6)
1096 006310 001365 BNE 6$
1097 006312 005726 TST (6)+
1098 006314 005037 001202 5$: CLR INTFLG ;CLEAR INTERRUPT FLAG
1099 006320 013777 001144 172512 4$: MOV BUF,@RSBA ;LOAD (CMA) BUSS ADDRESS
1100 006326 013702 001134 MOV WRDCT,R2 ;GET NEGATIVE
1101 006332 005402 NEG R2 ;WORD COUNT
1102 006334 010277 172476 MOV R2,@RSWC ;LOAD WC
    
```



CERSB-C RH70-RS03 DATA AND RELIABILITY TEST  
CERSBC.P11 14-AUG-78 08:29 TST4

C 4  
MACY11 30A(1052) 18-AUG-78 08:26 PAGE 28  
TEST FOR MULTI DISK MODE

SEQ 0041

1103	006340	032737	000400	001132
1104	006346	001021		
1105	006350	005737	001216	
1106	006354	001416		
1107	006356	032737	000040	001126
1108	006364	001403		
1109	006366	052737	001000	001176

BIT	#BIT8,FLAG	:RANDOM TEST?
BNE	1\$	:YES A PORT ONLY WITH NO MEM MGMT
TST	MMAVA	:MEM MGMT AVAILIABLE?
BEQ	1\$	:NO
BIT	#BIT5,FLAG2	:SET A17 IN RSCS1
BEQ	3\$	:NO
BIS	#BIT9,CMD	:YES

```

1110 006374 032737 000020 001126 3$: BIT #BIT4,FLAG2 ;SET A16?
1111 006402 001403 BEQ 1$ ;NO
1112 006404 052737 000400 001176 BIS #BIT8,CMD ;YES
1113 006422 113777 001176 172412 1$: MOV8 CMD,@RSCS1 ;LOAD FUNCTION REG.
1114 006420 000002 RTI ;RETURN FROM TRAP
1115
1116 ;RH70 DISK INTERRUPT HANDLER
1117 ;ROUTINE CONTINUES ON ERRORS
1118
1119 006422 042737 001000 001132 DKINT: BIC #BIT9,FLAG ;CLEAR ERROR BIT
1120 006430 005777 172376 TST @RSCS1 ;TEST FOR ERROR
1121 006434 100401 BMI 2$
1122 006436 000425 BR INTEXT ;JUMP IF NO ERRORS
1123 006440 017702 172366 2$: MOV @RSCS1,R2 ;GET CONTENTS OF CS1
1124 006444 042702 037777 BIC #37777,R2 ;CLEAR ALL BUT SC AND TRE
1125 006450 022702 140000 CMP #140000,R2 ;IS SC AND TRE BOTH SET?
1126 006454 001413 BEQ TRUERR ;YES THERE IS SOME KIND OF XFER ERROR
1127 006456 032777 100000 172360 BIT #100000,@RSDS ;IS THE ATA BIT SET?
1128 006464 001007 BNE TRUERR ;YES
1129 006466 104140 HLT !AS!DS ;WRONG UNIT INTERRUPTED
1130 ;IF YOU HAVE JUST POWERED UP A DRIVE OR
1131 ;ARE RUNNING THE POWER FAIL TEST,
1132 ;INTERRUPTS WILL OCCUR FROM DRIVES OTHER
1133 ;THAN THE UNIT UNDER TEST. IF THIS TYPEOUT
1134 ;SHOWS NO ERRORS IN THE REGISTERS OF THE DRIVE
1135 ;UNDER TEST, THAT DRIVE IS OK
1136 006470 012777 177777 172352 1$: MOV #-1,@RSAS ;CLEAR ALL ATA BITS
1137 006476 013716 001156 MOV HRDÉR,(SP) ;GET RETURN ADD.
1138 006502 000002 RTI ;RETRY
1139 006504 052737 001000 001132 TRUERR: BIS #BIT9,FLAG ;SET ERROR BIT
1140 006512 032737 004000 177570 INTEXT: BIT #BIT11,SWR ;HALT ON COMPLETION FLAG
1141 006520 001401 BEQ .+4
1142 006522 000000 HALT ;YES BIT 11 SET IN SWR HALT
1143 006524 032737 002000 001126 BIT #BIT10,FLAG2 ;WAIT IN BACKGROUND TEST?
1144 006532 001402 BEQ 1$ ;NO
1145 006534 012716 012264 2$: MOV @NPRRET,(SP) ;MODIFY RETURN ADD.
1146 006540 010637 001202 1$: MOV SP,INTFLG ;SET INT FLG
1147 006544 000002 RTI ;EXIT
1148 ;ROUTINE TO SET UP TRACK # FROM OPTION
1149 ;ENTER FROM JSR R5, OPDSEL
1150
1151 006546 032737 000040 001132 OPDSEL: BIT #BIT5,FLAG ;OPTIONAL DMA?
1152 006554 001403 BEQ 1$ ;NO
1153 006556 013737 001146 001140 MOV TDMA,DMA ;GET OPT. DMA
1154 006564 000205 1$: RTS R5 ;EXIT
    
```

1155	006566	005015	052123	051101	STABUF: .ASCIZ	<15><12>'STARTING 4K BANK #''
1156	006574	044524	043516	032040		
1157	006602	020113	040502	045516		
1158	006610	021440	000			
1159						
1160	006613	015	044012	053517	BUFSIZ: .ASCIZ	<15><12>'HOW MANY 4K BANKS? (OCTAL) ''
1161	006620	046440	047101	020131		
1162	006626	045464	041040	047101		
1163	006634	051513	020077	047450		
1164	006642	052103	046101	020051		
1165	006650	000				
1166						
1167	006651	015	052412	051116	UNRECO: .ASCIZ	<15><12>'UNRECOVERABLE''<15><12><12>
1168	006656	041505	053117	051105		
1169	006664	041101	042514	005015		
1170	006672	000012				
1171						
1172	006674	005015	047125	041101	NOWRIT: .ASCIZ	<15><12>'UNABLE TO WRITE''<15><12>
1173	006702	042514	052040	020117		
1174	006710	051127	052111	006505		
1175	006716	000012				
1176						
1177					.EVEN	
1178						
1179	006720	004737	014112		NOREC: JSR	PC,PRNT ;TYPEOUT?
1180	006724	001002			BNE	1\$ ;NO
1181	006726	104402	006651		TYPE	,UNRECO
1182	006732	005037	001204		1\$: CLR	LOPCNT ;CLEAR LOOP COUNTER
1183	006736	000207			RTS	PC
1184						
1185	006740	052737	040000	001126	FNDTYP: BIS	#BIT14,FLAG2 ;SET CHECK DRIVE TYPE FLAG
1186	006746	004737	001762		JSR	PC,UNTP ;CHECK DRIVE TYPE FLAG
1187	006752	042737	040000	001126	BIC	#BIT14,FLAG2 ;CLEAR DRIVE TYPE FLAG
1188	006760	000207			RTS	PC

```

1189          ;ROUTINE TO CALCULATE \ TURAL ADDR
1190
1191 006762 000302          PHYCOV: SWAB   R2          ;CALCULATE FROM PHYSICAL ADDR
1192 006764 004737 011566      JSR    PC,RRR2
1193 006770 006002          ROR    R2
1194 006772 042702 177770      BIC    #177770,R2          ;GET REG #
1195 006776 032777 000400 172026  BIT    #BIT8,@RSCS1        ;IS A16 SET?
1196 007004 001402          BEQ    1$                  ;NO
1197 007006 052702 000010      BIS    #BIT3,R2          ;YES
1198 007012 032777 001000 172012 1$: BIT    #BIT9,@RSCS1        ;IS A17 SET?
1199 007020 001402          BEQ    2$                  ;NO
1200 007022 052702 000020      BIS    #BIT4,R2          ;YES
1201 007026 013737 001074 001230 2$: MOV    STAMEM,WORK1      ;GET BANK # FOR -A- PORT
1202          ;*****
1203          ;*****
1204          ;*****
1205          ;*****
1206          ;*****
1207          ;*****
1208          ;*****
1209 007034 005737 001024          TST    TEMP2
1210 007040 001403          BEQ    3$
1211 007042 013737 001026 001230 3$: MOV    TEMP3,WORK1
1212 007050 163702 001230      SUB    WORK1,R2          ;GET STARTING BANK #
1213 007054 062702 000001      ADD    #1,R2            ;GET OFFSET FOR REG #
1214 007060 000302          SWAB   R2                ;GET BANK # INTO
1215 007062 006102          ROL    R2                ;UPPER BITS
1216 007064 006102          ROL    R2
1217 007066 006102          ROL    R2
1218 007070 006102          ROL    R2
1219 007072 006102          ROL    R2
1220 007074 017737 171740 001230  MOV    @RSBA,WORK1      ;GET OFFSET FOR ADDR IF ANY
1221 007102 162737 000002 001230  SUB    #2,WORK1        ;CORRECT IT
1222 007110 042737 160000 001230  BIC    #160000,WORK1    ;CLEAR JUNK
1223 007116 050237 001230      BIS    R2,WORK1        ;GET REG NO
1224 007122 013702 001230      MOV    WORK1,R2
1225 007126 000207          RTS    PC
1226
1227 007130 012777 006422 171730  VECTR: MOV    #DKINT,@RSVEC ;SETUP INTERRUPT VECTORS
1228 007136 013737 001072 177776  MOV    PRIORITY,PS     ;PRIORITY 4
1229 007144 000207          RTS    PC
1230
1231          ;THIS ROUTINE IS USED FOR DELAYING THE START OF THIS PROGRAM
1232          ;IF POWER FAILED DURING TESTING. THIS WILL GIVE THE DRIVES TIME TO GET UP
1233          ;TO SPEED. THE DELAY WILL BE ABOUT 3-5 MINUTES DEPENDING UPON THE PROCESSOR
1234
1235 007146 012737 177777 001226  TIMUP: MOV    #177777,WORK
1236 007154 012737 177777 001230 1$: MOV    #177777,WORK1
1237 007162 000240          2$: NOP
1238 007164 005337 001230      DEC    WORK1
1239 007170 001374          BNE    2$
1240 007172 005337 001226      DEC    WORK
1241 007176 001366          BNE    1$
1242 007200 000137 003176      JMP    ADTST
1243          ;ROUTINE TO SETUP DISK BUFFERS
1244          ;ADD WORD COUNT TO STARTING DISK ADDRESSES
    
```

```

1245          :COMPARE CALCULATED ADDRESS TO TERMINATING ADDRESS
1246
1247 007204 032737 000040 001132 DISBUF: BIT    #BITS,FLAG    ;DID OPERATOR SELECT PATTERNS
1248 007212 001402          BEQ    1$              ;NO
1249 007214 000137 007360          JMP    BUFEXIT      ;YES
1250 007220 004737 007550          JSR    PC,BLSZ     ;DEFINE BLOCK SIZE
1251 007224 013737 001160 001230          MOV    BLOCK,WORK1
1252 007232 005237 001140          INCSEC: INC    DMA    ;+1 SECTOR COUNT
1253 007236 022737 010000 001140          CMP    #10000,DMA  ;DONE YET?
1254 007244 001445          BEQ    BUFEXIT      ;YES
1255 007246 005337 001160          DEC    BLOCK        ;-1 FROM BLOCK COUNT
1256 007252 001401          BEQ    COMDAR       ;CMP DMA TO RSDA
1257 007254 000766          BR     INCSEC       ;RECYCLE
1258 007256 032737 001000 001132 COMDAR: BIT    #BIT9,FLAG  ;ANY ERRORS?
1259 007264 001401          BEQ    1$           ;NO ERRORS DO COMPARE ON RSDA
1260 007266 000207          RTS    PC           ;ERRORS DO NOT COMPARE RSDA
1261 007270 023777 001140 171544 $:    CMP    DMA,@RSDA    ;COMPARE RSDA WITH DMA
1262 007276 001425          BEQ    CMDAE        ;SHOULD BE EQUAL
1263 007300 104432          LOGR              ;AFTER TRANSFER RSDA AND DMA SHOULD BE
1264          ;IF NOT, RSDA IS NOT CORRECT. DMA CONTAINS
1265          ;WHAT RSDA SHOULD =
1266 007302 013701 001140          MOV    DMA,GOOD    ;GET DMA FOR CORRECT ANS IN GOOD
1267 007306 017700 171530          MOV    @RSDA,BAD  ;GET RSDA INTO BAD
1268 007312 104000          HLT                    ;RSDA=BAD DMA=GOOD SEE COMMENTS 7 LINES ABOVE
1269 007314 004737 014112          JSR    PC,PRNT     ;TYPEOUT?
1270 007320 001014          BNE    CMDAE        ;NO
1271 007322 011637 001226          MOV    (SP),WORK  ;GET TEST PC FROM WHERE IT CAME
1272 007326 104402 007332          TYPE    ,,+2      ;.ASCIZ '' TST PC=''
1273 007344 013746 001226          MOV    WORK,-(6)   ;PUT WORK ON STACK
1274 007350 104406          TYPES              ;TYPE STACK IN OCTAL - SUPRESS
1275 007352 105737 001132          CMDAE: TSTB    FLAG ;LAST DISK BUFFER?
1276 007356 100032          BPL    BUF INX     ;NO
    
```

1277	007360	005037	001140		BUFEXIT:	CLR	DMA		:CLEAR ADDRESS BITS LAST DISK BUFFER
1278	007364	062716	000002			ADD	#2,(6)		:INC STOCK POINTER
1279	007370	042737	000200	001132	AKH:	BIC	#200,FLAG		:CLEAR LAST DISK BUFFER FLAG
1280	007376	032737	000400	001132		BIT	#BIT8,FLAG		:RANDOM TEST OR ADDR TEST?
1281	007404	001404				BEQ	1\$		:NO
1282	007406	013737	001150	001134	2\$:	MOV	SWRDCT,WRDCT		
1283	007414	000454				BR	EXTDR		:EXIT
1284	007416	032737	000100	001132	1\$:	BIT	#BIT6,FLAG		:MULTI PORT?
1285	007424	001770				BEQ	2\$		:NO
1286	007426	005737	001120			TST	A0B1		:A OR B PORT?
1287	007432	001765				BEQ	2\$		:A PORT
1288	007434	013737	001116	001134		MOV	WDCTB,WRDCT		:B PORT
1289	007442	000441				BR	EXTDR		:GET OUT
1290	007444	005037	001232		BUFINX:	CLR	WORK2		:CLEAR WORK2 FOR BLOCK COUNTER
1291	007450	013702	001140			MOV	DMA,R2		:PUT WORKING DISK ADD INTO WORK
1292	007454	005237	001232		XINCSEC:	INC	WORK2		:INCREMENT BLOCK COUNT
1293	007460	022702	007777			CMP	#7777,R2		:CMP FOR LAST SECTOR
1294	007464	001405				BEQ	XINCSUR		:+1 SURFACE LAST SECTOR BRANCH
1295	007466	005202				INC	R2		:INC DMA
1296	007470	005337	001250			DEC	WORK1		:DEC BLOCK COUNT
1297	007474	001367				BNE	XINCSEC		:FILLED STANDARD BUFFER YET?
1298	007476	000734				BR	AKH		:WILL TAKE STANDARD SIZE WORD COUNT
1299	007500	013746	001232		XINCSUR:	MOV	WORK2,-(SP)		:SETTING UP BLOCK COUNT
1300	007504	000241				CLC			:FOR NON STANDARD BUFFER SIZE
1301	007506	006116				ROL	(SP)		
1302	007510	006116				ROL	(SP)		
1303	007512	006116				ROL	(SP)		
1304	007514	006116				ROL	(SP)		
1305	007516	006116				ROL	(SP)		
1306	007520	006116				ROL	(SP)		
1307	007522	012637	001134			MOV	(SP)+,WRDCT		
1308	007526	005737	001172			TST	RS04DI		:RS04?
1309	007532	001402				BEQ	1\$		:NO
1310	007534	006137	001134			ROL	WRDCT		:YES
1311	007540	052737	000200	001132	1\$:	BIS	#200,FLAG		:SET LAST DISK BUFFER FLAG
1312	007546	000207			EXTDR:	RTS	PC		:EXIT

```

1313          :THIS ROUTINE CONVERTS A WORD COUNT TO A BLOCK COUNT
1314
1315 007550 012737 000177 001160 BLSZ:  MOV    #177,BLUCK    :SETUP FOR RS04
1316 007556 005737 001172          TST     RS04DT      :RS04?
1317 007562 001003          BNE     2$          :YES
1318 007564 012737 000077 001160 1$:  MOV    #77,BLOCK    :PUT SECTOR SIZE INTO BLOCK
1319 007572 013702 001134          2$:  MOV    WRDCT,R2    :FETCH WORD COUNT
1320 007576 033702 001160          BIT     BLOCK,R2    :ARE THEY EQUAL?
1321 007602 001406          BEQ    RORBLK      :YES
1322 007604 043702 001160          BIC    BLOCK,R2    :SET UP BLOCK OVERFLOW
1323 007610 005237 001160          INC    BLOCK
1324 007614 063702 001160          ADD    BLOCK,R2
1325 007620 000241          RORBLK: CLC
1326 007622 006002          ROR    R2
1327 007624 006002          ROR    R2
1328 007626 004737 011566          JSR    PC,RRR2
1329 007632 005737 001172          TST    RS04DT      :RS04?
1330 007636 001401          BEQ    1$          :NO
1331 007640 006002          ROR    R2          :YES
1332 007642 010237 001160 1$:  MOV    R2,BLOCK    :BLOCK COUNT
1333 007646 000207          RTS     PC          :EXIT
1334
1335          :ROUTINE TO SELECT DATA PATTERNS FOR TEST
1336
1337          :ENTER FROM JSR R5,PASEL
1338
1339 007650 012737 010344 000004 PASEL: MOV    @MEM,@#4    :SETUP TRAP
1340 007656 012737 000340 000006          MOV    #340,@#6   :VECTOR
1341 007664 013700 001142          MOV    PATNU,R0    :SET UP PATTERN NUMBER
1342 007670 010603          MOV    R0,R3      :GET PATTERN #
1343 007672 000241          CLC              :MAKE IT =
1344 007674 006003          ROR    R3        :TO PATTERN # IN LISTING
1345 007676 010337 177570          MOV    R3,@NDISPL :DISPLAY PATTERN #
1346 007702 013737 001134 001226          MOV    WRDCT,WORK  :SET UP WORK
1347 007710 013701 001122          MOV    VADDR,R1   :LOC. OF OUTBUFFER
1348 007714 022700 000042 1$:  CMP    #42,R0     :TEST FOR RANDOM DATA NUMBER
1349 007720 001433          BEQ    RANDOM     :GO GENERATE RANDOM DATA
1350 007722 032737 000004 001132          BIT    @BIT2,FLAG :MAX TST?
1351 007730 001404          BEQ    2$        :NO
1352 007732 016037 000300 017326          MOV    PAT0(0),OUTBUF :GET PATTERN
1353 007740 000205          RTS     R5
1354 007742 016000 000300 2$:  MOV    PAT0(0),R0
1355          :XYZ*****?*****
1356          :*****
1357          :*****
1358          :XYZ*****?*****
1359          :*****
1360          :*****
1361          :XYZ*****?*****
1362          :*****
1363 007746 004737 017372          JSR    PC,MMPSET  :PRESET MEMORY MANAGEMENT REG
1364 007752 005737 001024          TST    TEMP2     :IS IT A TEST 2
1365 007756 001402          BEQ    FILBUF    :BRANCH IF NOT TEST 2
1366 007760 004537 017330          FILDAT: JSR    R5,MMPUSE :DO NEXT INSTRUCTION WITH MM
1367 007764 010021          FILBUF: MOV    RC,(1)+ :FILL BUFFER
1368 007766 005337 001226          DEC    WORK      :DEC. WORK COUNT
    
```

CERSB-C RH70-RS03 DATA AND RELIABILITY TEST  
CERSBC.P11 14-AUG-78 08:29 TST4

J 4  
MACY11 30A(1052) 18-AUG-78 08:26 PAGE 35  
TEST FOR MULTI DISK MODE

SEQ 0048

1369	007772	001372				BNE	FILDAT	:LOAD NEXT WORD
1370	007774	012737	000006	000004	PASEX:	MOV	#6,@#4	:RESTORE
1371	010002	005037	000006			CLR	@#6	:TRAP
1372	010006	000205				RTS	R5	:BUFFER FULL



```
1373 :RANDOM DATA GENERATOR SUBROUTINE
1374
1375 010010 013737 010204 010210 RANDOM: MOV LONUM,LOSAB
1376 010016 013737 010206 010212      MOV HINUM,HISAB
1377 :XYZ*****?*****
1378 :*****
1379 :*****
1380 :XYZ*****?*****
1381 :*****
1382 :*****
1383 :XYZ*****?*****
1384 :*****
1385 010024 004737 017372      JSR PC,MMPSET      :PRESET MEMORY MANAGEMENT REG
1386 010030 013700 010204      RAND1: MOV LONUM,R0      :SET UP R0 WITH 5 DIGITS LOW
1387 010034 013704 010206      MOV HINUM,R4      :SET UP R1 WITH 5 DIGITS HIGH
1388 010040 012703 000007      MOV #7,R3        :SET UP SHIFT COUNT
1389 010044 005002      CLR R2          :CLEAR R2
1390 010046 006300      SHIFT: ASL R0     :SHIFT R0 LEFT AND
1391 010050 006104      ROL R4         :ROTATE CARRY INTO LSB OF R1 INTO
1392 010052 006102      ROL R2         :ROTATE CARRY OUT OF R1 INTO R2
1393 010054 005303      DEC R3        :DECREMENT R3
1394 010056 001373      BNE SHIFT     :CONTINUE SHIFT LOOP
1395 010060 063700 010204      ADD LONUM,R0   :ADDN IN NUMBER TO MAKE X 129
1396 010064 005504      ADC R4        :PROPOGATE CARRY
1397 010066 063704 010206      ADD HINUM,R4   :ADDN IN NUMBER TO MAKE X 129
1398 010072 005502      ADC R2        :PROPOGATE CARRY
1399 010074 062700 001057      ADD #1057,R0  :ADDN LOW CONSTANT
1400 010100 005504      ADC R4        :PROPOGATE CARRIES
1401 010102 005502      ADC R2        :PROPOGATE AGAIN
1402 010104 062704 047401      ADD #47401,R4 :ADDN HIGH CONSTANT
1403 010110 005502      ADC R2        :PROPOGATE CARRY
1404 010112 062702 000006      ADD #6,R2     :ADDN HIGHEST CONSTANT
1405 010116 062700 000002      ADD #2,R0    :REPRIME R0 WITH HIGH DIGIT
1406 010122 005504      ADC R4        :PROPOGATE CARRY
1407 010124 010037 010204      MOV R0,LONUM  :PUT R0 BACK IN LONUM
1408 :XYZ*****?*****
1409 :*****
1410 :*****
1411 :XYZ*****?*****
1412 :*****
1413 :*****
1414 :XYZ*****?*****
1415 :*****
1416 010130 005737 001024      TST TEMP2     :IS IT A TEST 2
1417 010134 001402      BEQ 1$       :BRANCH IF NOT TEST 2
1418 010136 004537 017330      JSR R5,MMPUSE :DC NEXT INSTRUCTION WITH MM
1419 010142 010021      1$: MOV R0,(1)+  :HOLD LONUM FOR PROGRAM
1420 010144 005337 001226      DEC WORK
1421 010150 001413      BEQ EXGEN
1422 010152 010437 010206      MOV R4,HINUM  :PUT R1 BACK IN HINUM
1423 :XYZ*****?*****
1424 :*****
1425 :*****
1426 :XYZ*****?*****
1427 :*****
1428 :*****
```

```

1429
1430
1431 010156 005737 001024
1432 010162 001402
1433 010164 004537 017330
1434 010170 010421
1435 010172 005337 001226
1436 010176 001314
1437 010200 000137 007774
1438 010204 000000
1439 010206 000000
1440 010210 000000
1441 010212 000000
1442
1443 010214 013737 001220 001150
1444 010222 013737 001150 001134
1445 010230 000205

:XYZ*****?*****
:*****
TST TEMP2 ;IS IT A TEST 2
BEQ 2$ ;BRANCH IF NOT TEST 2
JSR R5,MMUSE ;DO NEXT INSTRUCTION WITH MM
2$: MOV R4,(1)+ ;HOLD HINUM FOR PROGRAM
DEC WORK
BNE RAND1
EXGEN: JMP PASEX ;RETURN TO PROGRAM
LONUM: 0
HINUM: 0
LOSAY: 0
HISAV: 0
RESTOR: MOV SAVWC,SWRDCT ;RESTORE ORIGINAL
MOV SWRDCT,WRDCT ;WORD COUNT
RTS R5
    
```

```

1446          :RANDOM DATA GENERATOR SUBROUTINE
1447          :WHEN SWITCH = 0 WE COME HERE
1448
1449 010232 013700 010340  RAND:  MOV     LONUM1,R0      ;SET UP R0 WITH 5 DIGITS LOW
1450 010236 013704 010342  MOV     HINUM1,R4      ;SET UP R1 WITH 5 DIGITS HIGH
1451 010242 012703 000007  MOV     #7,R3          ;SET UP SHIFT COUNT
1452 010246 005002          CLR     R2              ;CLEAR R2
1453 010250 006300          SHIFT1: ASL    R0          ;SHIFT R0 LEFT AND
1454 010252 006104          ROL    R4              ;ROTATE CARRY INTO LSB OF R1 INTO
1455 010254 006102          ROL    R2              ;ROTATE CARRY OUT OF R1 INTO R2
1456 010256 005303          DEC    R3              ;DECREMENT R3
1457 010260 001373          BNE    SHIFT1          ;CONTINUE SHIFT LOOP
1458 010262 063700 010340  ADD     LONUM1,R0      ;ADDN IN NUMBER TO MAKE X 129
1459 010266 005504          ADC    R4              ;PROPOGATE CARRY
1460 010270 063704 010342  ADD     HINUM1,R4      ;ADDN IN NUMBER TO MAKE X 129
1461 010274 005502          ADC    R2              ;PROPOGATE CARRY
1462 010276 062700 001057  ADD     #1057,R0        ;ADDN LOW CONSTANT
1463 010302 005504          ADC    R4              ;PROPOGATE CARRIES
1464 010304 005502          ADC    R2              ;PROPOGATE AGAIN
1465 010306 062704 047401  ADD     #47401,R4      ;ADDN HIGH CONSTANT
1466 010312 005502          ADC    R2              ;PROPOGATE CARRY
1467 010314 062702 000006  ADD     #6,R2           ;ADDN HIGHEST CONSTANT
1468 010320 062700 000002  ADD     #2,R0           ;REPRIME R0 WITH HIGH DIGIT
1469 010324 005504          ADC    R4              ;PROPOGATE CARRY
1470 010326 010037 010340  MOV     R0,LONUM1      ;PUT R0 BACK IN LONUM
1471 010332 010437 010342  MOV     R4,HINUM1      ;PUT R1 BACK IN HINUM
1472 010336 000205          EXGEN1: RTS           ;RETURN TO PROGRAM
1473 010340 000000          LONUM1: 0
1474 010342 000000          HINUM1: 0
1475
1476          :TRAP OUT ROUTINE WHEN CREATING DATA BUFFER
1477
1478 010344          MEM:
1479 010344 104402 010350          TYPE
1480 010364 012737 000006 000004 4$:  MOV     #6,@#4        ;.ASCIZ <15><12>'NON-X-MEM'
1481 010372 005037 000006          CLR    @#6            ;RESTORE
1482 010376 032737 100000 177570 BIT    #BIT15,SWR      ;TRAP
1483 010404 001401          BEQ   2$              ;HALT?
1484 010406 000000          HALT
1485 010410 000137 001234          2$:  JMP     @#BEGIN      ;NO
    
```

```
1486 ;THIS ROUTINE COMPARES THE DATA READ AGAINST THE DATA EXPECTED.  
1487 ;ALL ERRORS ARE REPORTED TO THE OPERATOR. IF BIT 4 OF THE SWITCH  
1488 ;REGISTER IS SET, THIS ROUTINE WILL CONTINUE COMPARING AFTER AN ERROR HAS BEEN  
1489 ;FOUND AND WILL REPORT UP TO 8 VERIFY ERRORS WITHIN THE SAME INPUT OPERATION.  
1490  
1491 010414 012737 177770 001152 COMPAR: MOV #10,ERCOUNT ;ERROR RETRY COUNTER  
1492 ;XYZ*****?  
1493 ;*****  
1494 ;*****  
1495 ;XYZ*****?  
1496 ;*****  
1497 ;*****  
1498 ;XYZ*****?  
1499 ;*****  
1500 010422 004737 017372 JSR PC,MMPSET ;PRESET MEMORY MANAGEMENT REG  
1501 010426 052737 000010 001126 BIS #BIT3,FLAG2 ;DOING COMPARE  
1502 010434 013737 001134 001232 MOV WRDCT,WORK2 ;GET THE WORD COUNT  
1503 010442 013737 001122 001154 MOV VADDR,SAVE ;SET UP OUTBUFFER POINTER  
1504 010450 005037 001200 CLR SWITCH ;CLEAR RANDOM PATTERN FLAG  
1505 010454 013737 010210 010340 MOV LOSAV,LONUM1 ;GET RANDOM BASE NOS.  
1506 010462 013737 010212 010342 MOV HISAV,HINUM1  
1507 010470 005737 001142 TST PATNU ;TEST FOR PATTERN 0  
1508 010474 001017 BNE 1$ ;NO  
1509 010476 005037 001226 CLR WORK ;CLEAR COUNTER  
1510 010502 062737 000001 001226 2$: ADD #1,WORK ;INC COUNTER  
1511 010510 001005 BNE 3$ ;INTERRUPT YET?  
1512 010512 104402 000510 TYPE ,NOINT  
1513 010516 104054 HLT !DA!WC!DS  
1514 010520 000137 001234 JMP @MBEGIN  
1515 010524 005737 001202 3$: TST INTFLG ;TEST FOR INT  
1516 010530 001764 BEQ 2$ ;WAIT FOR INT BEFORE COMPARING  
1517 010532 000426 BR CMPLP1 ;CONT  
1518 010534 022737 000042 001142 1$: CMP #42,PATNU ;IS THIS RANDOM PATTERN?  
1519 010542 001022 BNE CMPLP1 ;BRANCH IF YES  
1520 010544 005737 001202 CMPLP: TST INTFLG ;INTERRUPT YET?  
1521 010550 001775 BEQ CMPLP ;NO WAIT  
1522 010552 005737 001200 TST SWITCH  
1523 010556 001007 BNE 2$  
1524 010560 004537 010232 JSR R5,RAND  
1525 010564 013701 010340 MOV LONUM1,GOOD ;GET EVEN RANDOM WORD  
1526 010570 010637 001200 MOV SP,SWITCH ;SET RANDOM PATTERN FLAG  
1527 010574 000411 BR WRDCMP  
1528 010576 005037 001200 2$: CLR SWITCH  
1529 010602 013701 010342 MOV HINUM1,GOOD  
1530 010606 000404 BR WRDCMP  
1531 010610 013700 001142 CMPLP1: MOV PATNU,R0  
1532 010614 016001 000300 MOV PATO(R0),GOOD  
1533 ;XYZ*****?  
1534 ;*****  
1535 ;*****  
1536 ;XYZ*****?  
1537 ;*****  
1538 ;*****  
1539 ;XYZ*****?  
1540 ;*****  
1541 010620 005737 001024 WRDCMP: TST TEMP2
```

1542	010624	001406			BEQ	1\$	
1543	010626	042737	160000	001154	BIC	#160000,SAVE	:CLEAR PAR REG
1544	010634	052737	060000	001154	BIS	#60000,SAVE	:SET PAR 3
1545	010642	160177	170306		1\$: SUB	GOOD,@SAVE	:COMPARE DATA
1546	010646	001037			BNE	WDERR	:WORD IN ERROR
1547	010650	005337	001232		WRDINC: DEC	WORK2	:DECREMENT THE WORD COUNT
1548	010654	001430			BEQ	ADAM	:EXIT ROUTINE IF ZERO
1549	010656	062737	000002	001154	ADD	#2,SAVE	:UPDATE PATTERN ADDRESS
1550					:XYZ*****?		
1551					:*****		
1552					:*****		
1553					:XYZ*****?		
1554					:*****		
1555					:*****		
1556					:XYZ*****?		
1557					:*****		
1558	010664	005737	001024		TST	TEMP2	
1559	010670	001415			BEQ	1\$	
1560	010672	032737	100000	001154	BIT	#100000,SAVE	:IS 4K DONE
1561	010700	001411			BEQ	1\$	:BRANCH IF 4K NOT DONE
1562	010702	042737	100000	001154	BIC	#100000,SAVE	:CLEAR PAR BITS
1563	010710	052737	060000	001154	BIS	#60000,SAVE	:SET PAR 3
1564	010716	062737	000200	172346	ADD	#200,KIPAR3	:UPDATE PAR 3 TO NEXT 4K
1565	010724	022737	000042	001142	1\$: CMP	#42,PATNU	:IS THIS RANDOM PATTERN
1566	010732	001704			BEQ	CMPLP	:BRANCH IF YES
1567	010734	000731			BR	WRDCMP	:COMPARE NEXT WORD
1568	010736	042737	000010	001126	ADAM: BIC	#BIT3,FLAG2	:DONE WITH COMPARE
1569	010744	000205			RTS	R5	:EXIT THIS ROUTINE

1570	010746	005737	001202	WDERR:	TST	INTFLG		:DID INTERRUPT OCCUR YET?
1571	010752	001722			BEQ	WRDCMP		:BRANCH IF NO
1572	010754	032737	000100	177570	BIT	#BIT6,SWR		:TRY ALL?
1573	010762	001006			BNE	10\$		:YES
1574	010764	005737	001204		TST	LOPCNT		:FIRST READ ERROR?
1575	010770	001403			BEQ	10\$		:YES
1576	010772	005777	170034		TST	@RSCS1		:ANY ERRORS?
1577	010776	100757			BMI	ADAM		:YES DO NOT COMPARE
1578	011000	060177	170150	10\$:	ADD	GOOD,@SAVE		
1579	011004	017700	170144		MOV	@SAVE,BAD		:GET GOOD DATA
1580	011010	104436			LOGC			:LOG COMPARE ERROR
1581	011012	032737	001000	177570	BIT	#BIT9,SWR		:LOOP ON ERROR?
1582	011020	001401			BEQ	11\$		:NO
1583	011022	005726			TST	(6)+		:YES UPDATE SP
1584	011024	004737	014112	11\$:	JSR	PC,PRNT		:TYPEOUT?
1585	011030	001011			BNE	3\$		:NO
1586	011032	104402	011036		TYPE	..+2		:.ASCIZ <15><12>'COMPARE ERR'
1587	011054	104000		3\$:	HLT			:DATA COMPARE ERROR
1588	011056	004737	014112		JSR	PC,PRNT		:HAD TO DO IT THIS WAY SO
1589	011062	001022			BNE	13\$		:PROGRAM COULD LOOP ON ERROR
1590	011064	104402	011070		TYPE	..+2		:.ASCIZ ' ADDR='
1591	011100	005737	001216		TST	#MAVA		:IS MEM MGMT ON?
1592	011104	001406			BEQ	12\$		:NO
1593	011106	013746	177776		MOV	PS,-(6)		:GET PS
1594	011112	013746	001154		MOV	SAVE,-(6)		:GET VIRTUAL ADDR
1595	011116	104412			TYPEA			:CONVERT TO PHY AND TYPE
1596	011120	000403			BR	13\$		:CONT
1597	011122	013746	001154	12\$:	MOV	SAVE,-(6)		:GET ADDR
1598	011126	104406			TYPES			:TYPE IT
1599	011130	005037	001160	13\$:	CLR	BLOCK		:CLEAR THE BLOCK COUNTER
1600	011134	013702	001134		MOV	WRDCT,R2		:GET THE WORD COUNT
1601	011140	005202			INC	R2		:CORRECT FOR DA CALCULATIONS
1602	011142	163702	001232		SUB	WORK2,R2		:DETERMINE DISTANCE OF FAILURE INTO BUFFER
1603	011146	005737	001172	2\$:	TST	RS04DI		:RS04?
1604	011152	001403			BEQ	7\$		:NO
1605	011154	162702	000200		SUB	#200,R2		:RS03
1606	011160	000402			BR	9\$		:CONT
1607	011162	162702	000100	7\$:	SUB	#100,R2		
1608	011166	100403		9\$:	BMI	8\$		
1609	011170	005237	001160		INC	BLOCK		:UPDATE BLOCK COUNT FOR EACH 400 WORDS
1610	011174	000764			BR	2\$		

```

1611 011176 005737 001172      8$:  TST      RS04DT      ;RS04?
1612 011202 001403              BEQ      4$           ;NO
1613 011204 062702 000200      ADD      #200,R2     ;RS04
1614 011210 000402              BR       6$           ;CONT
1615 011212 062702 000100      4$:  ADD      #100,R2   ;RESTORE POSITIVE NUMBER
1616 011216 013737 001140 001230 6$:  MOV      DMA,WORK1 ;GET HEAD AND SECTOR ADDRESS
1617 011224 063737 001160 001230 5$:  ADD      BLOCK,WORK1
1618 011232 004737 014112      JSR      PC,PRNT     ;TYPEOUT?
1619 011236 001014              BNE     1$           ;NO
1620 011240 104402 011244      TYPE    ..+2        ;.ASCIZ '' DA=''
1621 011252 013746 001230      MOV     WORK1,-(6)  ;PUT WORK1 ON STACK
1622 011256 104406              TYPES   ;TYPE STACK IN OCTAL - SUPPRESS
1623 011260 104402 011264      TYPE    ..+2        ;.ASCIZ <15><12>
1624 011270 032737 000020 177570 1$:  BIT     #BIT4,SWR   ;RETRY?
1625 011276 001405              BEQ     ;NO
1626 011300 005237 001152      INC     ERCOUNT     ;UPDATE ERROR COUNTER
1627 011304 001402              BEQ     CLEAR
1628 011306 000137 010650      JMP     WRDINC
1629 011312 032737 000004 001132 CLEAR: BIT     #BIT2,FLAG ;XFER TEST?
1630 011320 001404              BEQ     ;NO
1631 011322 032737 010000 001132 BIT     #BIT12,FLAG ;READ?
1632 011330 001421              BEQ     2$           ;NO
1633 :XYZ*****?*****
1634 :*****
1635 :*****
1636 :XYZ*****?*****
1637 :*****
1638 :*****
1639 :XYZ*****?*****
1640 :*****
1641 011332 013701 001122      3$:  MOV     VADDR,R1   ;GET STARTING ADD OF BUFFER
1642 011336 013700 001134      MOV     WRDCT,R0    ;NOW
1643 011342 004737 017372      JSR     PC,MMSET    ;PRESET MM
1644 011346 005737 001024      TST     TEMP2       ;IS IT TEST 2
1645 011352 001402              BEQ     1$           ;BRANCH IF NOT TEST 2
1646 011354 004537 017330      4$:  JSR     R5,MMUSE  ;DO NEXT INSTRUCTION WITH MM
1647 011360 005021              1$:  CLR     (R1)+     ;CLEAR BUFFER
1648 011362 005300              DEC     R0           ;COUNT LOCATIONS
1649 011364 001373              BNE     4$           ;WAIT TILL DONE
1650 011366 042737 000010 001126 BIC     #BIT3,FLAG2 ;DONE WITH COMPARE
1651 011374 000205              2$:  RTS     R5        ;NOW GET OUT
1652
1653 011376 013737 001076 017326 APORT: MOV     SAVAST,OUTBUF ;SET STARTING ADDR FOR OUTBUF
1654 011404 013737 001076 001122 MOV     SAVAST,VADDR ;SAVE OUTBUF ADDR
1655 011412 005737 001216      TST     MMVA        ;MEM MGMT?
1656 011416 001411              BEQ     EXTT        ;NO
1657 011420 013702 001104      MOV     SAVMGA,R2   ;SET UP MEM MGMT
1658 011424 004737 011650      MMSET: JSR     PC,STMM2 ;SETUP MEM MGMT
1659 011430 010237 001122      MOV     R2,VADDR
1660 011434 013737 001124 017326 MOV     PHADDR,OUTBUF
1661 011442 000207      EXTT:  RTS     PC
    
```

```

1662                                     :TYPE CAN NOT WRITE BLOCK
1663
1664 011444 004737 014112 WTNO: JSR PC,PRNT ;TYPEOUT?
1665 011450 001002 BNE 1$ ;NO
1666 011452 104402 000674 TYPE ,NOWRIT
1667 011456 005037 001204 1$: CLR LOPCNT ;CLEAR ERR COUNTER
1668 011462 000207 RTS PC
1669
1670                                     :ROUTINE TO SET UP STARTING ADDRESS FOR ALL PORTS
1671                                     :AND TO CREATE WORD COUNT MAX= 20K
1672
1673 011464 013702 001074 EXTMEM: MOV STAMEM,R2 ;GET BANK #
1674 011470 005702 TST R2 ;DID HE TYPE 0?
1675 011472 001001 BNE 3$ ;NO
1676 011474 005202 INC R2 ;YES MAKE 1
1677 011476 005737 001216 3$: TST MMVA
1678 011502 001006 BNE 1$ ;BRANCH IF MEM MGMT AVAILABLE
1679 011504 000241 CLC
1680 011506 004737 011566 JSR PC,RRR2
1681 011512 010237 001076 MOV R2,SAVAST ;SAVE A STARTIND ADDR
1682 011516 000404 BR 2$ ;GET WC
1683 011520 000302 1$: SWAB R2
1684 011522 006002 ROR R2
1685 011524 010237 001104 MOV R2,SAVMGA ;SAVE ADDR FOR A PORT
1686 011530 013702 001112 2$: MOV SIZEAP,R2 ;GET 4K BLOCK COUNT
1687 011534 005202 INC R2
1688 011536 163702 001074 SUB STAMEM, R2 ;LIMIT WC TO MEMORY SIZE
1689 011542 000241 8$: CLC
1690 011544 006002 ROR R2 ;NO CONVERT TO WC
1691 011546 004737 011566 JSR PC,RRR2
1692 011552 042702 000077 BIC #77,R2 ;CLEAR BLOCK COUNT
1693 011556 010237 001150 MOV R2,SURDCT ;SAVE -A- PORT WC
1694 011562 000400 BR 7$ ;CONT
1695                                     :4$:MOV#60000,SURDCT;MAKE 20K
1696 011564 000207 7$: RTS PC
1697
1698 011566 006002 RRR2: ROR R2
1699 011570 006002 ROR R2
1700 011572 006002 ROR R2
1701 011574 006002 ROR R2
1702 011576 000207 RTS PC
1703
1704 011600 032737 000200 177570 WATT: BIT #BIT7,SWR ;WAIT IN BACKGROUND?
1705 011606 001003 BNE 1$ ;NO
1706 011610 004737 012154 JSR PC,XWAIT ;YES
1707 011614 000401 BR 2$ ;CONT
1708 011616 000001 1$: WAIT
1709 011620 000207 2$: RTS PC
1710
1711 011622 004737 014112 TYPREC: JSR PC,PRNT ;TYPEOUT?
1712 011626 001007 BNE 1$ ;NO
1713 011630 104402 000702 TYPE ,RECOV
1714 011634 013746 001204 MOV LOPCNT,-(6) ;GET COUNT
1715 011640 104406 TYPES ;TYPE IT
1716 011642 104402 000757 TYPE ,PLFLF
1717 011646 000207 1$: RTS PC
    
```



```

1718
1719 011650 005737 001216 STMM2: TST MMAVA ;MEM MGMT?
1720 011654 001002 BNE 3$ ;YES
1721 011656 000137 012142 JMP MDON ;GET OUT
1722 011662 005037 172340 3$: CLR @#KIPAR0
1723 011666 010237 001154 MOV R2,SAVE ;SAVE R2
1724 011672 010237 172342 MOV R2,@#KIPAR1
1725 011676 006302 ASL R2 ;CALCULATE PHYSICAL ADDR
1726 011700 006302 ASL R2
1727 011702 006302 ASL R2
1728 011704 006302 ASL R2
1729 011706 006302 ASL R2 ;THIS BIT IS A17
1730 011710 042737 000040 001126 BIC #BIT5,FLAG2 ;CLEAR A17?
1731 011716 103003 BCC 1$ ;SET A17
1732 011720 052737 000040 001126 BIS #BIT5,FLAG2 ;SET BIT 5 FOR A17
1733 011726 042737 000020 001126 1$: BIC #BIT4,FLAG2 ;CLEAR A16 FLAG
1734 011734 006302 ASL R2 ;GET A16 BIT
1735 011736 103003 BCC 2$ ;CLEAR A16
1736 011740 052737 000020 001126 BIS #BIT4,FLAG2 ;SET FLAG FOR A16
1737 011746 010237 001124 2$: MOV R2,PHADDR ;GET PHYSICAL ADDR
1738 011752 013702 001154 MOV SAVE,R2 ;SET UP MEM MGMT
1739 011756 062702 000200 ADD #200,R2
1740 011762 010237 172344 MOV R2,@#KIPAR2
1741 011766 062702 000200 ADD #200,R2
1742 011772 010237 172346 MOV R2,@#KIPAR3
1743 011776 062702 000200 ADD #200,R2
1744 012002 010237 172350 MOV R2,@#KIPAR4
1745 012006 062702 000200 ADD #200,R2
1746 012012 010237 172352 MOV R2,@#KIPAR5
1747 012016 062702 000200 ADD #200,R2
1748 012022 010237 172354 MOV R2,@#KIPAR6
1749 012026 012737 077406 172300 MOV #200*256.-400+UP+RW,@#KIPDR0 ;SET KIPDR0=RW UP 200 BLOCKS
1750 012034 012737 077406 172302 MOV #200*256.-400+UP+RW,@#KIPDR1 ;SET KIPDR1=RW UP 200 BLOCKS
1751 012042 012737 077406 172304 MOV #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
1752 012050 012737 077406 172306 MOV #200*256.-400+UP+RW,@#KIPDR3 ;SET KIPDR3=RW UP 200 BLOCKS
1753 012056 012737 077406 172310 MOV #200*256.-400+UP+RW,@#KIPDR4 ;SET KIPDR4=RW UP 200 BLOCKS
1754 012064 012737 077406 172312 MOV #200*256.-400+UP+RW,@#KIPDR5 ;SET KIPDR5=RW UP 200 BLOCKS
1755 012072 012737 077406 172314 MOV #200*256.-400+UP+RW,@#KIPDR6 ;SET KIPDR6=RW UP 200 BLOCKS
1756 012100 012737 077406 172316 MOV #200*256.-400+UP+RW,@#KIPDR7 ;SET KIPDR7=RW UP 200 BLOCKS
1757 012106 012737 177600 172356 MOV #177600,@#KIPAR7
1758 012114 012702 020000 MOV #20000,R2
1759 012120 012737 012144 000250 MOV #MABT0,@#MVEC
1760 012126 012737 000020 172516 MOV #20,SR3 ;TURN ON 22 BIT MODE
1761 012134 012737 000001 177572 MOV #1,@#SRO ;TURN ON MEM MGMT
1762 012142 000207 MDON: RTS PC
1763 :MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
1764 012144 104402 000541 MABT0: TYPE ,MTRAP
1765 012150 000000 HALT
1766 012152 000002 RTI ;CAUSED THE ABORT
  
```

```

767                                     :BACKGROUND TEST FOR INTERRUPTS
1768
1769 012154 052737 002000 001126 XWAIT: BIS #BIT10,FLAG2 ;WAITING IN BACKGROUND TEST
1770 012162 012737 070000 012274 MOV #70000,NPRCNT ;SETUP TIMEOUT COUNTER
1771 012170 012701 012277 MOV #NPR1+1,R1 ;SETUP WAIT LOOP
1772 012174 112711 000200 MOVB #200,(R1)
1773 012200 2$:
1774 012200 105421 NEGB (R1)+
1775 012202 105441 NEGB -(R1)
1776 012204 105421 NEGB (R1)+
1777 012206 105441 NEGB -(R1)
1778 012210 105421 NEGB (R1)+
1779 012212 105441 NEGB -(R1)
1780 012214 105421 NEGB (R1)+
1781 012216 105441 NEGB -(R1)
1782 012220 105421 NEGB (R1)+
1783 012222 105441 NEGB -(R1)
1784 012224 105421 NEGB (R1)+
1785 012226 105441 NEGB -(R1)
1786 012230 105421 NEGB (R1)+
1787 012232 105441 NEGB -(R1)
1788 012234 105421 NEGB (R1)+
1789 012236 105441 NEGB -(R1)
1790 012240 102401 BVS 1$
1791 012242 000000 HALT ;ARITHMETIC OPERATION FAILED RUN DIAG
1792 012244 005337 012274 1$: DEC NPRCNT
1793 012250 001353 BNE 2$
1794 012252 104402 000510 TYPE ,NOINT
1795 012256 104054 HLT !DA!WC!DS
1796 012260 000137 001234 JMP @BEGIN
1797 012264 042737 002000 001126 NPRRET: BIC #BIT10,FLAG2 ;CLEAR BKGROUND FLG
1798 012272 000207 RTS PC
1799 012274 000000 NPRCNT: 0
1800 012276 000000 NPR1: 0
1801 ;CLEAR ERROR TABLE
1802
1803 012300 012704 000020 ERRCL: MOV #20,R4 ;CLEAR
1804 012304 012703 017266 MOV #ERTAB,R3 ;ERROR
1805 012310 005023 1$: CLR (R3)+ ;TABLE
1806 012312 005304 DEC R4 ;DONE YET?
1807 012314 001375 BNE 1$ ;NO
1808 012316 005037 001004 CLR PCNT ;CLEAR
1809 012322 005037 001006 CLR PCNT+2 ;PASS COUNT
1810 012326 005037 001130 CLR DROP ;CLEAR ALL DROPPED DRIVES
1811 012332 000205 RTS R5 ;RETURN
  
```

```

1812 :RH70 POWER FAIL TEST #1
1813 :THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 270.
1814 :A MESSAGE WILL BE TYPED OUT 'LOAD SW WITH UNIT # AND CONT..'
1815 :THE OPERATOR NOW HAS TO LOAD THE UNIT # IN OCTAL INTO THE SW REGISTER
1816 :IN BITS 00-01-AND 02. THEN HIT CONT. THE PROGRAM WILL
1817 :WRITE THE COMPLETE DISK WITH A 125252 PATTERN. THE PROGRAM WILL THEN
1818 :TELL OPERATOR TO POWER DOWN. UNTIL THE POWER FAIL, THE PROGRAM WILL
1819 :CONTINUE WRITING THE SAME PATTERN ON THE DISK.
1820 :WHEN POWER FAIL OCCURS THE TRANSFER IS ABORTED
1821 :AND THE PROGRAM HALTS. THE OPERATOR SHOULD
1822 :NOW TURN POWER BACK ON. THE PROGRAM RESTARTS AND CHECKS FOR WRITE ERRORS.
1823 :ONLY ONE ERROR IS ACCEPTABLE. THAT ERROR MAY BE AN OPI (BIT13 RSER)OR A DCK
1824 : (BIT 15 RSER). IF THESE ARE THE ONLY ERRORS THAT OCCUR, THE DRIVE IS OK.
1825 :IF NO ERRORS OCCUR, THE PROGRAM WILL TYPE OUT 'OK'.
1826 :THE PROGRAM WILL THEN TELL YOU WHEN TO POWER DOWN AGAIN
1827
1828 .***ONLY ONE ERROR IS CONSIDERED ACCEPTABLE***
1829
1830 012334 012706 000500 PFT1: MOV #500,SP ;SET UP STACK
1831 012340 005037 001024 CLR TEMP2
1832 012344 104402 000561 TYPF ,LOADSW
1833 012350 000000 HALT
1834 012352 004737 007130 JSR PC,VECTR ;SETUP INT VECTOR
1835 012356 013737 177570 001164 MOV @SWR,UNNUM ;SAVE IT
1836 012364 004737 006740 JSR PC,FNDTYP ;TST FOR RS03 OR 04
1837 012370 104426 PFWATT: CLRDV ;CLEAR ALL REG
1838 012372 004737 013152 JSR PC,POWFAL ;WRITE 125252 ON DISK
1839 012376 005037 001140 PFWAT: CLR DMA
1840 012402 012737 012624 000024 MOV #DOWN,24 ;SET UP POWER FAIL VEC.
1841 012410 012737 000340 000026 MOV #340,26
1842 012416 012737 000161 001176 MYBYWR: MOV #161,CMD ;WRITE WITH I/E
1843 012424 104416 DKCMD ;DO IT
1844 012426 004737 011600 JSR PC,WATT ;WAIT FOR INTERRUPT
1845 012432 032737 001000 001132 3$: BIT #BIT9,FLAG ;ANY ERRORS?
1846 012440 001406 BEQ 1$ ;NO
1847 012442 104006 HLT !DA!DB
1848 012444 012777 177777 166376 MOV #-1,@RSAS ;CLEAR ALL
1849 012452 005077 166370 CLR @RSER ;ERRORS
1850 012456 004737 007204 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
1851 012462 000755 BR MYBYWR
1852 012464 000744 BR PFWAT
  
```

```

1853 012466 012737 012474 001156 UPCHK: MOV #1$,HRDER ;RETURN HERE IF WRONG DRIVE INTERRUPTS
1854 012474 005037 001140 1$: CLR DMA
1855 012500 104426 CLRDV ;INIT DRIVE
1856 012502 013737 001072 177776 CHKDAT: MOV PRIORITY,PS
1857 012510 012737 000151 001176 MOV #151,CMD ;WRITECHECK WITH I/E
1858 012516 104416 DKCMD ;DO IT
1859 012520 013737 001072 177776 MOV PRIORITY,PS
1860 012526 004737 011600 JSR PC,WAIT ;WAIT FOR INTERRUPT
1861 012532 032737 001000 001132 3$: BIT #BIT9,FLAG ;ANY ERRORS?
1862 012540 001411 BEQ 1$ ;NO
1863 012542 104006 HLT !DB.DA
1864 012544 052737 100000 001126 BIS #BIT15,FLAG2 ;SET ERROR FLAG
1865 012552 005077 166270 CLR @RSER ;CLEAR ALL
1866 012556 012777 177777 166264 MOV #-1,@RSAS ;ERRORS
1867 012564 004737 007204 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
1868 012570 000744 BR CHKDAT
1869 012572 005737 001126 TST FLAG2 ;ANY ERRORS?
1870 012576 100405 BMI 2$ ;YES
1871 012600 104402 012604 TYPE ..+2 ;.ASCIZ <15><12>'DK'
1872 012612 042737 100000 001126 2$: BIC #BIT15,FLAG2 ;CLEAR ERROR FLAG
1873 012620 000137 012370 JMP PFWAIT ;GO WAIT FOR ANOTHER
1874
1875
1876 ;POWER DOWN ROUTINE - ABORT DISK AND HALT
1877
1878 012624 012737 012634 000024 DOWN: MOV #UPP,24 ;SET POWER FAIL VECTOR
1879 012632 000000 HALT
1880
1881 012634 012737 012624 000024 UPP: MOV #DOWN,24
1882 012642 012706 000500 MOV #500,SP
1883 012646 013777 001164 166160 MOV UNNUM,@RSCS2 ;GET UNIT #
1884 012654 032777 000200 166162 1$: BIT #BIT7,@RSDS ;WAIT FOR DRIVE READY
1885 012662 001774 BEQ 1$
1886 012664 000137 012466 JMP UPCHK ;GO CHECK DISK
    
```

```

1887 :POWER FAIL TEST #2
1888 :THIS TEST WILL TEST THE SAME DRIVE THAT WAS TESTED IN THE 1ST POWER FAIL TEST
1889 :THE PROGRAM WILL WRITE THE COMPLETE DISK WITH A 125252 PATTERN AND WILL
1890 :THEN TELL THE OPERATOR TO POWER DOWN THE PROCESSOR.
1891 :THE PROGRAM WILL THEN WRITE CHECK THE DISK WHILE WAITING FOR A POWER FAIL.
1892 :WHEN THE POWER FAIL OCCURS, THE WRITE CHECKING IS ABORTED AND
1893 :THE PROCESSOR WILL HALT.
1894 :THE OPERATOR SHOULD THEN TURN POWER BACK ON, THE PROGRAM WILL
1895 :START WRITE CHECKING THE DISK AGAIN
1896 :***NO ERRORS SHOULD OCCUR.***
1897 :THE PROGRAM WILL TYPE OUT 'OK' IF NO ERRORS OCCUR.
1898 :THE PROGRAM WILL THEN TELL YOU TO POWER DOWN.
1899 :DO NOT POWER OFF THE PROCESSOR AGAIN UNTIL THE PROGRAM TELLS YOU SO.
1900
1901 012670 C12706 000500 PFT2: MOV #500,SP ;SET UP STACK
1902 012674 005037 001024 CLR TEMP2
1903 012700 042737 001000 001126 BIC #BIT9,FLAG2 ;CLEAR POWER FAIL
1904 012706 012737 012730 001156 MOV #PWRFL,HRDR ;RETURN HERE IF WRONG DRIVE INT.
1905 012714 104426 CLRDV ;INIT DRIVE
1906 012716 004737 007130 JSR PC,VECTRR ;SETUP INT VECTOR
1907 012722 004737 013152 PWRFL2: JSR PC,POWFAL ;WRITE 125252 ON DISK
1908 012726 000401 BR PWRFL ;WRITE CHECK
1909 012730 104426 CLRDV ;INIT DRIVE
1910 012732 005037 001140 PWRFL: CLR DMA
1911 012736 012737 013102 000024 MOV #PWRDN,24 ;SET UP POWER FAIL VEC.
1912 012744 012737 000340 000026 MOV #340,26
1913 012752 013737 001072 177776 CHKDSK: MOV PRIORITY,PS ;ENABLE I/E
1914 012760 012737 000151 001176 MOV #151,CMD ;WRITE CHECK WITH I/E
1915 012766 104416 DKCMD ;DO IT
1916 012770 004737 011600 JSR PC,WATT ;WAIT FOR INTERRUPT
1917 012774 032737 001000 001132 3$: BIT #BIT9,FLAG ;ANY ERRORS?
1918 013002 001411 BEQ 1$ ;NO
1919 013004 104002 HLT !DB ;YES
1920 013006 052737 100000 001126 BIS #BIT15,FLAG2 ;SET ERROR FLAG
1921 013014 005077 166026 CLR @RSER ;CLEAR ALL
1922 013020 012777 177777 166022 MOV #-1,@RSAS ;ERRORS
1923 013026 004737 007204 1$: JSR PC,DISBUF ;CHECK NEXT BUFFER
1924 013032 000747 BR CHKDSK
1925 013034 032737 001000 001126 BIT #BIT9,FLAG2 ;DID POWER FAIL?
1926 013042 001733 BEQ PWRFL ;NO
1927 013044 005737 001126 TST FLAG2 ;ANY ERRORS?
1928 013050 100405 BMI 2$ ;YES
1929 013052 104402 013056 TYPE ;.ASCIZ <15><12>'OK''
1930 013064 042737 100000 001126 2$: BIC #BIT15,FLAG2 ;CLEAR ERRORS
1931 013072 042737 001000 001126 BIC #BIT9,FLAG2 ;CLEAR POWER FAIL FLAG
1932 013100 000710 4$: BR PWRFL2

```

```

1933 ;ROUTINE TO ABORT DISK DURING POWER FAIL
1934
1935 013102 012737 013112 000024 PWRDN: MOV #PWRUP,24 ;SET UP RESTART
1936 013110 000000 HALT
1937
1938 013112 012737 013102 000024 PWRUP: MOV #PWRDN,24 ;RESET POWER FAIL VECTOR
1939 013120 012706 000500 MOV #500,SP
1940 013124 013777 001164 165702 MOV UNNUM,@RSCS2 ;GET UNIT #
1941 013132 052737 001000 001126 BIS #BIT9,FLAG2 ;SET POWER FAIL BIT
1942 013140 032777 000200 165676 1$: BIT #BIT7,@RSDS ;WAITING FOR
1943 013146 001774 BEQ 1$ ;DRIVE READY
1944 013150 000667 BR PWRF1 ;GO CHECK DISK
1945
1946
1947 ;ROUTINE TO WRITE THE COMPLETE DISK
1948 ;WITH 125252 PATTERN
1949 ;WRITE CHECK AND REPORT ERRORS IF THEY OCCUR
1950 ;REPORT 'OK' AT COMPLETION
1951
1952 013152 012737 000020 001142 POWFAL: MOV #20,PATNU ;SET UP PATTERN
1953 013160 042737 000004 001132 BIC #BIT2,FLAG ;CLEAR XFER MODE FLAG
1954 013166 052737 010000 001126 BIS #BIT12,FLAG2
1955 013174 005037 001140 CLR DMA
1956 013200 012737 020000 017326 MOV #20000,OUTBUF ;GET STARTING ADDR FOR BUF
1957 013206 012737 020000 001122 MOV #20000,VADDR
1958 013214 012737 010000 001150 MOV #10000,SWRDCT ;SETUP WORD COUNT
1959 013222 013737 001150 001134 MOV SWRDCT,WRDCT
1960 013230 005037 001120 CLR AOB1 ;A PORT ONLY
1961 013234 013737 017326 001144 MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
1962 013242 004537 007650 JSR R5,PASEL ;GENERATE DATA BUFFER
1963 013246 012737 000161 001176 WRDNU: MOV #161,CMD ;WRITE WITH I/E
1964 013254 104416 DKCMD ;DO IT
1965 013256 004737 011600 JSR PC,WATT ;WAIT FOR INTERRUPT
1966 013262 012737 000151 001176 2$: MOV #151,CMD ;WRITECHECK I/E
1967 013270 104416 DKCMD ;DO IT
1968 013272 004737 011600 JSR PC,WATT ;WAIT FOR INTERRUPT
1969 013276 032737 001000 001132 4$: BIT #BIT9,FLAG ;ANY ERRORS?
1970 013304 001402 BEQ 1$ ;NO
1971 013306 104006 HLT !DB!DA ;YES
1972 013310 000000 HALT ;CAN NOT WRITE WITHOUT ERROR
1973 013312 004737 007204 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
1974 013316 000753 BR WRDNU ;WRITE NEW BUFFER
1975 013320 104402 TYPE ;.ASCIZ <15><12>'POWER DOWN'
1976 013342 000207 RTS PC
  
```

1977	013344	032737	000010	177570	OUT:	BIT	#BIT3,SWR	;TYPEOUT ERROR COUNT?
1978	013352	001532				BEQ	1\$	;NO
1979	013354	005004				CLR	R4	;CLEAR UNIT #
1980	013356	005003				CLR	R3	
1981	013360	053737	001130	001166		BIS	DROP,UNITSV	;RESTORE ALL DRIVES
1982	013366	013737	001166	001226		MOV	UNITSV,WORK	;GET UNITS ON SYSTEM
1983	013374	012705	000401			MOV	#401,R5	;SETUP TEST FOR UNITS
1984	013400	030537	001226		4\$:	BIT	R5,WORK	;IS THIS UNIT ON SYS
1985	013404	001006				BNE	2\$	;YES
1986	013406	005204			5\$:	INC	R4	;INC UNIT #
1987	013410	010403				MOV	R4,R3	;SAVE UNIT #
1988	013412	000241				CLC		
1989	013414	006105				ROL	R5	;GET NEXT DRIVE
1990	013416	103505				BCS	3\$	;DONE
1991	013420	000767				BR	4\$	;FIND NEXT DRIVE
1992	013422				2\$:			
1993	013422	104402	013426			TYPE	..+2	;;ASCIZ <15><12>'UNIT ''
1994	013436	010446				MOV	R4,-(6)	;PUT R4 ON STACK
1995	013440	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
1996	013442	004737	013720			JSR	PC,GETERR	;GET ERROR COUNT
1997	013446	010304				MOV	R3,R4	;RESTORE UNIT #
1998	013450	104402	013454			TYPE	..+2	;;ASCIZ <15><12>
1999	013460	104402	000630			TYPE	..WRERR	
2000	013464	104402	013470			TYPE	..+2	;;ASCIZ 'S ''
2001	013474	013746	001206			MOV	WRITER,-(6)	;PUT WRITER ON STACK
2002	013500	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2003	013502	104402	013506			TYPE	..+2	;;ASCIZ <15><12>
2004	013512	104402	000657			TYPE	..RDERR	
2005	013516	104402	013522			TYPE	..+2	;;ASCIZ 'S ''
2006	013526	013746	001212			MOV	READER,-(6)	;PUT READER ON STACK
2007	013532	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2008	013534	104402	013540			TYPE	..+2	;;ASCIZ <15><12>
2009	013544	104402	000642			TYPE	..WCKERR	
2010	013550	104402	013554			TYPE	..+2	;;ASCIZ 'S ''
2011	013560	013746	001210			MOV	WCERR,-(6)	;PUT WCERR ON STACK
2012	013564	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2013	013566	104402	013572			TYPE	..+2	;;ASCIZ <15><12>'COMPARE ERRS ''
2014	013612	013746	001214			MOV	COMERR,-(6)	;PUT COMERR ON STACK
2015	013616	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2016	013620	104402	013624			TYPE	..+2	;;ASCIZ <15><12>
2017	013630	000666				BR	5\$	;GET NEXT DRIVE
2018	013632	043737	001130	001166	3\$:	BIC	DROP,UNITSV	;REDROP DRIVES
2019	013640	062706	000002		1\$:	ADD	#2,SP	;RESTORE SP DUE TO JMP EXIT FROM JSR ROUTINE
2020	013644	005137	001120			COM	A0B1	;SET A OR B PORT FLAG
2021	013650	032737	000040	177570		BIT	#BIT5,SWR	;TYPEOUT PASS COUNT?
2022	013656	001035				BNE	DONE	;NO
2023	013660	104402	013664			TYPE	..+2	;;ASCIZ <15><12>'END PASS ''
2024	013700	013746	001006			MOV	PCNT+2,-(6)	;PUT PCNT+2 ON STACK
2025	013704	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2026	013706	104402	013712			TYPE	..+2	;;ASCIZ <15><12>
2027	013716	000415				BR	DONE	

```

2028 013720 006304          GETERR: ASL      R4          ;GET LOC IN
2029 013722 006304          ASL      R4          ;ERR TABLE
2030 013724 062704 017266    ADD      #ERTAB,R4
2031 013730 112437 001206    MOVB    (R4)+,WRITER ;GET WRITE ERRS
2032 013734 112437 001212    MOVB    (R4)+,READER ;GET READ ERRS
2033 013740 112437 001210    MOVB    (R4)+,WCERR  ;GET WRITE CK ERRS
2034 013744 112437 001214    MOVB    (R4)+,COMERR ;GET COMPARE ERRS
2035 013750 000207          RTS      PC
2036
2037          .SBTTL          $DONE - BELL AND SCOPE ROUTINE
2038
2039 013752 104400          DONE:   SCOPE          ;TERMINATING SCOPE FOR LOOPING
2040 013754 062737 000001 001006  ADD      #1,PCNT+2    ;ADD 1 TO THE PASS COUNT
2041 013762 005537 001004          ADC      PCNT          ;MAKE IT DOUBLE PREC.
2042 013766 013700 000042          4$:    MOV      @#42,R0 ;GET MONITOR ADDRESS
2043 013772 001405          BEQ     $END1         ;IF NONE
2044 013774 000005          RESET
2045 013776 004710          SENDAD: JSR     7,(0) ;GO TO MONITOR
2046 014000 000240 000240 000240  JMO     240,240,240 ;SAVE ROOM FOR ACT11
2047 014006 000137 003176          SEND1: JMP     JMO,ADTST ;RETURN
2048
2049 014012 000000          .TBIT: 0              ;T BIT FLAG
2050
2051 014014 012702 000001          .LOGW: MOV      #1,R2   ;LOG WRITE ERR
2052 014020 005003          CLIND: CLR     R3      ;CLEAR INDEX FOR TABLE
2053 014022 000413          BR
2054
2055 014024 012702 000400          .LOGR: MOV      #400,R2 ;LOG WRITE ERR
2056 014030 000773          BR      CLIND
2057
2058 014032 012702 000001          .LOGWC: MOV     #1,R2   ;LOG WRITC CK ERR
2059 014036 012703 000002          SETIND: MOV    #2,R3   ;SET INDEX FOR NEXT WD
2060 014042 000403          BR      ADDR
2061
2062 014044 012702 000400          .LOGC: MOV      #400,R2 ;LOG COMPARE ERR
2063 014050 000772          BR      SETIND
2064
2065 014052 005737 001204          ADDR:  TST     LOPCNT   ;1ST TIME ERROR?
2066 014056 001014          BNE     1$           ;NO DO NOT COUNT IT
2067 014060 013704 001164          MOV     UNUM,R4     ;GET UNIT #
2068 014064 006304          ASL     R4          ;GET
2069 014066 006304          ASL     R4          ;POSITION IN
2070 014070 060304          ADD     R3,R4       ;ERR TABLE
2071 014072 060264 017266    ADD     R2,ERTAB(R4) ;TO ADD ERROR
2072 014076 004737 014112    JSR     PC,PRNT     ;TYPEOUT?
2073 014102 001402          BEQ     1$          ;YES
2074 014104 004737 014760          JSR     PC,DRP     ;SHOULD I DROP DRIVE?
2075 014110 000002          1$:    RTI
2076
2077 014112 032737 020000 177570 PRNT:  BIT     #BIT13,SWR ;INHIBIT TYPEOUT?
2078 014120 000207          RTS      PC

```



2079	014122	052737	000004	001126	RSREG:	BIS	#BIT2,FLAG?	:SET ERROR FLAG
2080	014130	005737	016134			TST	.HLTCT	:SHOULD WE TYPE GOOD AND BAD
2081	014134	001017				BNE	8\$	:NO
2082	014136	104402	014142			TYPE	..+2	:.ASCIZ 'BAD='
2083	014150	010046				MOV	BAD,-(6)	:PUT BAD ON STACK
2084	014152	104404				TYPEO		:TYPE STACK IN OCTAL
2085	014154	104402	014160			TYPE	..+2	:.ASCIZ 'GOOD='
2086	014170	010146				MOV	GOOD,-(6)	:PUT GOOD ON STACK
2087	014172	104404				TYPEO		:TYPE STACK IN OCTAL
2088	014174				8\$:			
2089	014174	104402	014200			TYPE	..+2	:.ASCIZ 'CS1='
2090	014206	017746	164620			MOV	@RSCS1,-(6)	:PUT @RSCS1 ON STACK
2091	014212	104404				TYPEO		:TYPE STACK IN OCTAL
2092	014214				1\$:			
2093	014214	104402	014220			TYPE	..+2	:.ASCIZ 'ER='
2094	014226	017746	164614			MOV	@RSER,-(6)	:PUT @RSER ON STACK
2095	014232	104404				TYPEO		:TYPE STACK IN OCTAL
2096	014234				2\$:			
2097	014234	104402	014240			TYPE	..+2	:.ASCIZ 'CS2='
2098	014246	017746	164562			MOV	@RSCS2,-(6)	:PUT @RSCS2 ON STACK
2099	014252	104404				TYPEO		:TYPE STACK IN OCTAL
2100	014254	104402	014260			TYPE	..+2	:.ASCIZ '<15><12>'
2101	014264	104402	014270			TYPE	..+2	:.ASCIZ 'CS3='
2102	014276	017746	164562			MOV	@RSCS3,-(6)	:PUT @RSCS3 ON STACK
2103	014302	104404				TYPEO		:TYPE STACK IN OCTAL
2104	014304	104402	014310			TYPE	..+2	:.ASCIZ 'BAE='
2105	014316	017746	164540			MOV	@RSBAE,-(6)	:PUT @RSBAE ON STACK
2106	014322	104404				TYPEO		:TYPE STACK IN OCTAL
2107	014324	104402	014330			TYPE	..+2	:.ASCIZ '<15><12>'
2108	014334	032737	000200	016134		BIT	#200,.HLTCT	:PRINT SECOND SET ?
2109	014342	001112				BNE	SEEC	:YES
2110	014344	032737	000100	016134		BIT	#AS,.HLTCT	:PRINT ER ?
2111	014352	001410				BEQ	3\$	:NO
2112	014354	104402	014360			TYPE	..+2	:.ASCIZ 'AS='
2113	014366	017746	164456			MOV	@RSAS,-(6)	:PUT @RSAS ON STACK
2114	014372	104404				TYPEO		:TYPE STACK IN OCTAL
2115	014374	032737	000020	016134	3\$:	BIT	#BA,.HLTCT	:PRINT BUS ADDRESS
2116	014402	001410				BEQ	4\$	:NO
2117	014404	104402	014410			TYPE	..+2	:.ASCIZ 'BA='
2118	014416	017746	164416			MOV	@RSBA,-(6)	:PUT @RSBA ON STACK
2119	014422	104404				TYPEO		:TYPE STACK IN OCTAL
2120	014424	032737	000004	016134	4\$:	BIT	#DA,.HLTCT	:PRINT DA ?
2121	014432	001410				BEQ	5\$	:NO
2122	014434	104402	014440			TYPE	..+2	:.ASCIZ 'DA='
2123	014446	017746	164370			MOV	@RSDA,-(6)	:PUT @RSDA ON STACK
2124	014452	104404				TYPEO		:TYPE STACK IN OCTAL
2125	014454	032737	000010	016134	5\$:	BIT	#WC,.HLTCT	:PRINT WC?
2126	014462	001410				BEQ	6\$	:NO
2127	014464	104402	014470			TYPE	..+2	:.ASCIZ 'WC='
2128	014476	017746	164334			MOV	@RSWC,-(6)	:PUT @RSWC ON STACK
2129	014502	104404				TYPEO		:TYPE STACK IN OCTAL
2130	014504	032737	000040	016134	6\$:	BIT	#DS,.HLTCT	:DRIVE STATUS
2131	014512	001410				BEQ	9\$	:NO
2132	014514	104402	014520			TYPE	..+2	:.ASCIZ 'DS='
2133	014526	017746	164312			MOV	@RSDS,-(6)	:PUT @RSDS ON STACK
2134	014532	104404				TYPEO		:TYPE STACK IN OCTAL

CERSB-C RH70-RS03 DATA AND RELIABILITY TEST  
CERSBC.P11 14-AUG-78 08:29

MACY11 30A(1052) <sup>B 6</sup> 18-AUG-78 08:26 PAGE 53  
\$DONE - BELL AND SCOPE ROUTINE

SEQ 0066

2135 014534 032737 000002 016134 9\$:

BIT #DB..HLTCT ;PRINT DATA BUFFER

2136	014542	001461				BEQ	PTDONE		:NO
2137	014544	104402	014550			TYPE	..+2		:.ASCIZ '' DB=''
2138	014556	017746	164272			MOV	@RSDB,-(6)		:PUT @RSDB ON STACK
2139	014562	104404				TYPE0			:TYPE STACK IN OCTAL
2140	014564	000137	014706			JMP	PTDONE		:GET OUT
2141	014570	042737	000200	016134	SEEC:	BIC	#200,.HLTCT		:CLEAR COMMON BIT
2142	014576	032737	000240	016134		BIT	#DT,.HLTCT		:PRINT DRIVE TYPE?
2143	014604	001410				BEQ	10\$		:NO
2144	014606	104402	014612			TYPE	..+2		:.ASCIZ '' DT=''
2145	014620	017746	164234			MOV	@RSDT,-(6)		:PUT @RSDT ON STACK
2146	014624	104404				TYPE0			:TYPE STACK IN OCTAL
2147	014626	032737	000220	016134	10\$:	BIT	#MR,.HLTCT		:PRINT MR?
2148	014634	001410				BEQ	11\$		:NO
2149	014636	104402	014642			TYPE	..+2		:.ASCIZ '' MR=''
2150	014650	017746	164202			MOV	@RSMR,-(6)		:PUT @RSMR ON STACK
2151	014654	104404				TYPE0			:TYPE STACK IN OCTAL
2152	014656	032737	000204	016134	1\$:	BIT	#LA,.HLTCT		:PRINT LA?
2153	014664	001410				BEQ	PTDONE		:NO
2154	014666	104402	014672			TYPE	..+2		:.ASCIZ '' LA=''
2155	014700	017746	164146			MOV	@RSLA,-(6)		:PUT @RSLA ON STACK
2156	014704	104404				TYPE0			:TYPE STACK IN OCTAL
2157	014706	032737	010000	001126	PTDONE:	BIT	#BIT12,FLAG2		:POWER FAIL TEST?
2158	014714	001111				BNE	RETT		:YES
2159	014716	104402	014722			TYPE	..+2		:.ASCIZ <15><12>'PASS ''
2160	014732	013746	001006			MOV	PCNT+2,-(6)		:PUT PCNT+2 ON STACK
2161	014736	104406				TYPES			:TYPE STACK IN OCTAL - SUPPRESS
2162	014740	032737	001000	177570		BIT	#BIT9,SWR		:LOOPING ON ERROR?
2163	014746	001404				BEQ	DRP		:NO
2164	014750	104402	014754			TYPE	..+2		:.ASCIZ <15><12>
2165	014760	032737	000001	177570	DRP:	BIT	#BIT0,SWR		:DROP DRIVE?
2166	014766	001464				BEQ	RETT		:NO
2167	014770	013704	001164			MOV	UNNUM,R4		:GET UNIT #
2168	014774	004737	013720			JSR	PC,GETERR		:GET ERRORS
2169	015000	063737	001206	001212		ADD	WRITER,READER		:ADD THE ERRORS
2170	015006	063737	001212	001210		ADD	READER,WCERR		
2171	015014	063737	001210	001214		ADD	WCERR,COMERR		
2172	015022	022737	000023	001214		CMP	#23,COMERR		:DROP DRIVE?
2173	015030	103043				BHIS	RETT		:NO
2174	015032	053737	001170	001130		BIS	UNCMP,DROP		:DROP DRIVE
2175	015040	104402	015044			TYPE	..+2		:.ASCIZ <15><12>'DROPPED UNIT ''
2176	015064	013746	001164			MOV	UNNUM,-(6)		:PUT UNNUM ON STACK
2177	015070	104406				TYPES			:TYPE STACK IN OCTAL - SUPPRESS
2178	015072	104402	000757			TYPE	,CRLF		
2179	015076	113703	001130			MOVB	DROP,R3		:GET DROPPED UNITS
2180	015102	113704	001166			MOVB	UNITSV,R4		:GET ALL DRIVES
2181	015106	020304				CMP	R3,R4		:ALL DRIVES DROPPED?
2182	015110	001003				BNE	2\$		:NO
2183	015112	000000				HALT			:NO MORE DRIVES
2184	015114	000137	001234			JMP	@BEGIN		:RESTART TEST
2185	015120	032737	100000	001132	2\$:	BIT	#BIT15,FLAG		:DID OPERATOR SELECT PATTERN
2186	015126	001002				BNE	3\$		:YES
2187	015130	005037	001142			CLR	PATNU		:NO CLEAR IT
2188	015134	000137	006154		3\$:	JMP	@EXTPPR		:GET NEXT DRIVE
2189	015140	000207			RETT:	RTS	PC		

```

2190 ;ROUTINE TO RESTORE LOADER
2191 015142 013705 015166 RLDL: MOV LDR1,R5 ;GET FIRST ADDRESS OF WHERE LOADER IS
2192 ;TO BE RESTORED
2193 015146 012704 017500 MOV #17500,R4 ;ADDRESS WHERE LOADER IS STORED
2194 015152 012702 000155 MOV #155,R2 ;WORD COUNT
2195 015156 012425 1$: MOV (R4)+,(R5)+ ;RESTORE
2196 015160 005302 DEC R2
2197 015162 001375 BNE 1$
2198 015164 000000 HALT ;DONE
2199 015166 017500 LDR1: .WORD 17500 ;FIRST ADDRESS WHERE LOADERS ARE SAVED
2200
2201 PARCSR=172100
2202 000114 PARVEC=114
2203 015170 012737 015262 000114 .MAMK: MOV #,PARSRV,@PARVEC
2204 015176 012737 000340 000116 MOV #340,@PARVEC+2 ;SET PRI LEVEL TO 7
2205 015204 013746 000004 MOV @#4,-(SP) ;SAVE CURRENT ERROR VECTOR
2206 015210 013746 000006 MOV @#6,-(SP) ;SAVE PRIORITY LEVEL
2207 015214 012737 000006 000004 MOV #6,@#4
2208 015222 012737 000002 000006 MOV #RTI,@#6
2209 015230 012700 172100 MOV #PARCSR,R0 ;GET FIRST CSR ADDR
2210 015234 012702 000001 MOV #1,R2
2211 015240 012720 000001 1$: MOV #1,(R0)+ ;SET ACTION ENABLE IF AVAILABLE
2212 015244 006302 ASL R2 ;SHIFT AVAILABILITY INDICATOR
2213 015246 103374 BCC 1$
2214 015250 012637 000006 MOV (SP)+,@#6 ;RESTORE ERROR VECTOR PRIORITY
2215 015254 012637 000004 MOV (SP)+,@#4 ;AND INTERRUPT VECTOR
2216 015260 000207 RTS PC
2217 ;PARITY MEMORY TRAP
2218
2219 .PARSRV:
2220 015262 104402 015266 TYPE ..+2 ;.ASCIZ <15><12>'PARITY ERR'
2221 015304 052737 004000 001126 BIS #BIT11,FLAG2 ;SET ERROR FLAG
2222 015312 104402 000757 TYPE .CRLF
2223 015316 104402 015322 TYPE ..+2 ;.ASCIZ 'HIER= '
2224 015332 013746 177742 MOV #ERADD,-(SP)
2225 015336 104404 TYPE0
2226 015340 104402 015344 TYPE ..+2 ;.ASCIZ '' LOER=''
2227 015354 013746 177740 MOV #LERADD,-(SP)
2228 015360 104404 TYPE0
2229 015362 104402 015366 TYPE ..+2 ;.ASCIZ '' ME REG=''
2230 015400 013746 177744 MOV #MEMERR,-(SP)
2231 015404 104404 TYPE0
2232 015406 032737 000010 001126 BIT #BIT3,FLAG2 ;WERE WE COMPARING DURING ERROR?
2233 015414 001422 BEQ 13$ ;NO
2234 015416 104402 015422 TYPE ..+2 ;.ASCIZ '' ADDR=''
2235 015432 005737 001216 TST #MAVA ;IS MEM MGMT ON?
2236 015436 001406 BEQ 12$ ;NO
2237 015440 013746 177776 MOV #PS,-(6) ;GET PS
2238 015444 013746 001154 MOV #SAVE,-(6) ;GET VIRTUAL ADDR
2239 015450 104412 TYPEA ;CONVERT TO PHY AND TYPE
2240 015452 000403 BR 13$ ;CONT
2241 015454 013746 001154 12$: MOV #SAVE,-(6) ;GET ADDR
2242 015460 104406 TYPEA ;TYPE IT
2243 015462 032737 100000 177570 13$: BIT #BIT15,SWR ;HALT ON ERROR?
2244 015470 001401 BEQ 1$ ;NO
2245 015472 000000 HALT ;YES

```

CERSB-C RM70-RS03 DATA AND RELIABILITY TEST  
CERSBC.P11 14-AUG-78 08:29

MACY11 30A(1052) E 6 18-AUG-78 08:26 PAGE 56  
\$DONE - BELL AND SCOPE ROUTINE

SEQ 0069

2246 015474 012706 000500  
2247 015500 000137 003212

18:

MOV #500,SP  
JMP EXMFLG

;RESET STACK  
;RESTART TEST

```

2248 .SBTTL $TYPE - TTY TYPEOUT ROUTINE
2249
2250 ;THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
2251 ;CALL CAN BE IN ONE OF 3 FORMS. 1) 'TYPE ,ADR' - TYPES THE
2252 ;MESSAGE STARTING IN LOCATION 'ADR:' 2) 'TYPE ,CHAR' - TYPES
2253 ;THE ASCII 'CHAR', AND 3) 'PRINT <<15><12>'MESSAGE'> - TYPES
2254 ;THE MESSAGE WHICH IS INLINE ASCII. THE FILLER CHARACTER WHICH IS
2255 ;TYPED AFTER A LINE FEED IS IN FILCHR AND THE NUMBER OF FILLERS
2256 ;IS IN FILCHR+1.
2257
2258 015504 010446 .TYPE: MOV R4,-(6) ;SAVE R4
2259 015506 010546 MOV R5,-(6) ;SAVE R5
2260 015510 017605 000004 MOV @4(6),R5 ;GET ADDRESS TO BE TYPED
2261 015514 032705 177400 BIT #177400,R5 ;IS IT A TYPED?
2262 015520 001002 BNE 1$ ;NO
2263 015522 016605 000004 MOV 4(6),R5 ;GET ADDRESS OF CHARACTER
2264 015526 105715 $: TSTB (R5) ;TERMINATOR?
2265 015530 001423 BEQ 2$ ;GET OUT IF SO
2266 015532 122715 000012 CMPEB #12,(R5) ;IS THE CHAR A LINE FEED
2267 015536 001012 BNE 4$ ;NO - GET OUT
2268 015540 113704 001015 MOVB FILCHR+1,R4 ;GET THE FILL COUNT
2269 015544 113777 001014 163246 5$: MOVB FILCHR,@TPB ;TYPE A FILLER
2270 015552 105777 163240 TSTB @TPB ;DONE YET?
2271 015556 100375 BPL -4 ;NO - WAIT
2272 015560 005304 DEC R4 ;DEC COUNT
2273 015562 001370 BNE 5$ ;LOOP UNTIL 0
2274 015564 112577 163230 4$: MOVB (R5)+,@TPB ;LOAD AND TYPE THE CHARACTER
2275 015570 105777 163222 TSTB @TPB ;IS THE PRINTER READY
2276 015574 100375 BPL -4 ;WAIT UNTIL IT IS
2277 015576 000753 BR 1$ ;GET THE NEXT CHARACTER
2278 015600 017646 000004 2$: MOV @4(6),-(6) ;GET ADDRESS TO BE TYPED
2279 015604 062766 000002 000006 ADD #2,6(6) ;ADD 2 TO THE ADDRESS
2280 015612 022666 000004 CMP (6)+,4(6) ;IS IT .+2?
2281 015616 001006 BNE 3$ ;NO
2282 015620 062705 000002 ADD #2,R5 ;ADD 2 TO THE ADDRESS
2283 015624 042705 000001 BIC #1,R5 ;BACK UP TO AN EVEN BYTE
2284 015630 010566 000004 MOV R5,4(6) ;RESTORE ADDRESS
2285 015634 012605 3$: MOV (6)+,R5 ;RESTORE R5
2286 015636 012604 MOV (6)+,R4 ;RESTORE R4
2287 015640 000002 RTI ;RETURN

```

```

2288          .SBTTL          $SCOPE - SCOPE LOOP HANDLLR
2289
2290          ;THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
2291          ;LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
2292          ;"SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
2293          ;RECORDS THE STARTING ADDRESS OF THE SUBTEST IN 'LAD:'
2294
2295 015642 032737 040000 177570 .SCOPE: BIT      #SW14,@#SWR      ;LOOP ON TEST?
2296 015650 001045          BNE      .KIT          ;LOOP ON TEST IS SET
2297 015652 000416          BR       3$          ;SKIP - NOP FOR XOR TESTER
2298 015654 013746          MOV      @#4,-(6)      ;PUSH @#4 ON STACK
2299 015660 012737 015700 000004 MOV      #4$,@#4      ;SET FOR TIMEOUT
2300 015666 005737 177060          TST      @#177060      ;ERROR ON XOR?
2301 015672 012637 000004          MOV      (6)+,@#4      ;POP STACK INTO @#4
2302 015676 000422          BR       .SVLAD      ;NO ERROR - GO TO NEXT TEST
2303 015700 022626          4$: CMP      (6)+,(6)+      ;CLEAR STACK
2304 015702 012637 000004          MOV      (6)+,@#4      ;POP STACK INTO @#4
2305 015706 000426          BR       .KIT          ;ERROR - LOOP ON TEST
2306 015710 032737 004000 177570 3$: BIT      #SW11,@#SWR      ;KILL ITERATIONS
2307 015716 001012          BNE      .SVLAD      ;YES - KILL ITERATIONS
2308 015720 105737 001001          TSTB   ICNT+1          ;FIRST ONE?
2309 015724 001404          BEQ      2$          ;BRANCH IF FIRST
2310 015726 123737 016012 001001 CMPB    TIMES,ICNT+1      ;DONE?
2311 015734 003013          BGT      .KIT          ;BRANCH IF NOT
2312 015736 112737 000001 001001 2$: MOVB   #1,ICNT+1      ;FIRST ITERATION
2313 015744 105237 001000          .SVLAD: INCB   ICNT      ;COUNT TEST NUMBERS
2314 015750 011637 001010          MOV      (6),LAD      ;SAVE LOOP ADDRESS
2315 015754 013737 001000 177570 MOV      ICNT,@#DISPLAY ;DISPLAY TEST NO. AND ITERATION COUNT
2316 015762 000002          RTI          ;RETURN
2317
2318 015764 105237 001001          .KIT: INCB   ICNT+1      ;INC THE ITERATION COUNT
2319 015770 013737 001000 177570 .OVER: MOV      ICNT,@#DISPLAY ;SET UP DISPLAY
2320 015776 005737 001010          TST      LAD          ;FIRST ONE?
2321 016002 001760          BEQ      .SVLAD      ;YES
2322 016004 013716 001010          MOV      LAD,(6)      ;FUDGE RETURN ADDRESS
2323 016010 000002          RTI          ;FIXES PS
2324
2325 016012 000001          TIMES: 1          ;RUN 1 TIMES
  
```

```

2326          .SBTTL          $HLT - HLT ROUTINE (ERROR TYPEOUT)
2327
2328          ;THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
2329          ;ADDRESS OF THE 'HLT'. IT ALSO COUNTS THE NUMBER OF ERRORS
2330          ;AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
2331          ;'HALT' ON ERROR, AND INHIBIT TYPEOUTS. AN OPTIONAL ARGUMENT
2332          ;(HLT+3) WILL BE PLACED IN '.HLTCT:' FOR ADITIONAL TYPEOUTS.
2333
2334 016014 005237 001002          .HLT:  INC      ERRORS      ;INC THE ERROR COUNT
2335 016020 032737 020000 177570  BIT      #SW13,@#SWR  ;SKIP TYPEOUT IF SET
2336 016026 001025                BNE      2$          ;SKIP TYPEOUTS
2337 016030 104402 016034          TYPE     ..+2        ;.ASCIZ <15><12>
2338 016040 011637 001012          MOV      (6),HLTADR ;PUT ADDRESS OF INSTRUCTION ON STACK
2339 016044 162737 000002 001012  SUB      #2,HLTADR  ;FUDGE ADDRESS
2340 016052 117737 162734 016134  MOVB     @HLTADR,.HLTCT ;GET HLT ARGUEMENT
2341 016060 013746 001012          MOV      HLTADR,-(6) ;PUT HLTADR ON STACK
2342 016064 104404                TYPEO    ;TYPE STACK IN OCTAL
2343 016066 104402 016072          TYPE     ..+2        ;.ASCIZ ""
2344 016076 004737 014122          JSR      PC,RSREG   ;GO TO USER ERROR ROUTINE
2345 016102 005737 177570 2$:    TST      @#SWR     ;HALT ON ERROR
2346 016106 100001                BPL      .+4        ;SKIP IF CONTINUE
2347 016110 000000                HALT     ;HALT ON ERROR!
2348 016112 032737 001000 177570  BIT      #SW9,@#SWR  ;CHECK FOR INHIBIT LOOP ON ERROR
2349 016120 001003                BNE      3$          ;SKIP IF LOOP ON ERROR
2350 016122 105037 001001          CLRB    ICNT+1     ;CLEAR ITERATION COUNT
2351 016126 000002                RTI      ;RETURN
2352 016130 000137 015764 3$:    JMP      .K11      ;LOOP ON TEST UNTIL NO ERRORS
2353
2354 016134 000000          .HLTCT: 0          ;HLT ARGUMENT
  
```



```

2355 .SBTTL          SOCIAL - OCTAL TYPEOUT ROUTINE
2356
2357 ;THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
2358 ;ALL 6 CHARACTERS, SUPPRESS LEADING ZFROES, OR TYPE THE
2359 ;16 BITS. IT IS CALLED VIA THE TYOCT, TYPBIT, OR TYOCS MACRO'S.
2360
2361 016136 012737 170101 016324 .TYPEB: MOV      #170101,.PR      ;SET BIT FLAG AND 16. CHARACTER COUNT
2362 016144 000411                BR          .PTIT          ;NOW TYPE IT IN BIT FORM
2363 016146 112737 000001 016324 .TYPEO: MOVB   #1,.PR      ;SET ZERO FILL SWITCH
2364 016154 000402                BR          .+6           ;SKIP
2365 016156 005037 016324 .TYPES: CLR    .PR          ;SUPRESS LEADING ZERO'S
2366 016162 112737 177772 016325 MOVB      #-6,.PR+1      ;SET COUNT
2367 016170
2368 016170 010446                MOV      R4,-(6)          ;PUSH R4 ON STACK
2369 016172 010546                MOV      R5,-(6)          ;PUSH R5 ON STACK
2370 016174 016605 000010        MOV      10(6),R5        ;GET THE DATA
2371 016200 012704 016326        MOV      #,PR+2,R4      ;SET POINTER TO FIRST ASCII CHAR.
2372 016204 105014                CLRB     (4)            ;CLEAR FIRST BYTE
2373 016206 000411                BR       .PRF           ;ROTATE FIRST BIT
2374 016210 105014                .PRL:   CLRB    (4)      ;CLEAR BYTE OF CHARACTER
2375 016212 032737 000100 016324 BIT      #100,.PR       ;BIT TYPING MODE?
2376 016220 001004                BNE     .PRF           ;YES - SKIP 2 ROTATES
2377 016222 006105                ROL     R5             ;ROTATE BIT INTO C
2378 016224 106114                ROLB   (4)            ;PACK IT
2379 016226 006105                ROL     R5             ;ROTATE BIT INTO C
2380 016230 106114                ROLB   (4)            ;PACK IT
2381 016232 006105                .PRF:   ROL     R5             ;ROTATE BIT INTO C
2382 016234 106114                ROLB   (4)            ;PACK IT
2383 016236 105714                TSTB   (4)            ;IS IT ZERO?
2384 016240 001402                BEQ    .+6            ;SKIP INC
2385 016242 105237 016324        INCB   .PR            ;SET FILL SWITCH
2386 016246 105737 016324        TSTB   .PR            ;CHECK FILL SWITCH
2387 016252 001402                BEQ    .+6            ;SKIP BITSET
2388 016254 152724 000060        BISH   #'0,(4)+      ;MAKE INTO ASCII CHAR
2389 016260 105237 016325        INCB   .PR+1          ;INC COUNT
2390 016264 001351                BNE    .PRL           ;REPEAT
2391 016266 022704 016326        CMP    #,PR+2,R4      ;EMPTY BUFFER?
2392 016272 001002                BNE    .+6            ;SKIP IF NOT
2393 016274 112724 000060        MOVB   #'0,(4)+      ;LOAD 1 ZERO
2394 016300 105014                CLRB   (4)            ;NULL TERMINATOR
2395 016302 104402 016326        TYPE   ..PR+2          ;TYPE IT
2396 016306 012605                MOV    (6)+,R5        ;POP STACK INTO R5
2397 016310 012604                MOV    (6)+,R4        ;POP STACK INTO R4
2398 016312 016666 000002 000004 MOV    2(6),4(6)      ;GET RID OF
2399 016320 012616                MOV    (6)+,(6)      ;DATA WORD
2400 016322 000002                RTI                    ;RETURN
2401
2402 016324 000012                .PR:   .BLKW   '2      ;COUNT, SWITCH, AND OUTPUT BUFFER

```

```

2403          .SBTTL          $POWER - POWER DOWN AND UP ROUTINES
2404
2405          ;THIS IS THE POWER FAIL ROUTINE WHICH WILL SAVE ALL
2406          ;THE GENERAL REGISTERS AND USER DEFINED REGISTERS THEN
2407          ;WAIT FOR POWER TO GO DOWN AND BE RESTORED.
2408          ;IF THERE ISN'T ENOUGH TIME FOR SAVING ALL THE REGISTERS,
2409          ;THE PROGRAM WILL HALT AT '.ILLUP'.
2410
2411 016350 012777 016476 000126 .POWER: MOV      #.ILLUP,@.PUVEC ;SET FOR FAST UP
2412 016356 012777 000340 000122      MOV      #340,@.PUVEC$+2 ;PRIO:7
2413 016364 010046              MOV      R0,-(6) ;PUSH R0 ON STACK
2414 016366 010146              MOV      R1,-(6) ;PUSH R1 ON STACK
2415 016370 010246              MOV      R2,-(6) ;PUSH R2 ON STACK
2416 016372 010346              MOV      R3,-(6) ;PUSH R3 ON STACK
2417 016374 010446              MOV      R4,-(6) ;PUSH R4 ON STACK
2418 016376 010546              MOV      R5,-(6) ;PUSH R5 ON STACK
2419 016400 010637 016502      MOV      SP,.SAVR6 ;SAVE SP
2420 016404 012777 016414 000072      MOV      #.POWUP,@.PUVEC ;SET UP VECTOR
2421 016412 000000              HALT          ;WAIT FOR PF
2422
2423 016414 013706 016502      .POWUP: MOV      .SAVR6,SP ;GET SP
2424 016420 005001              CLR      R1 ;WAIT LOOP FOR THE TTY
2425 016422 005201      1$: INC      R1 ;WAIT FOR THE INC
2426 016424 001376              BNE     1$ ;OF WORD
2427 016426 012605              MOV      (6)+,R5 ;POP STACK INTO R5
2428 016430 012604              MOV      (6)+,R4 ;POP STACK INTO R4
2429 016432 012603              MOV      (6)+,R3 ;POP STACK INTO R3
2430 016434 012602              MOV      (6)+,R2 ;POP STACK INTO R2
2431 016436 012601              MOV      (6)+,R1 ;POP STACK INTO R1
2432 016440 012600              MOV      (6)+,R0 ;POP STACK INTO R0
2433 016442 012737 016350 000024      MOV      #.POWER,@#24 ;SET UP THE POWER DOWN VECTOR
2434 016450 012737 000340 000026      MOV      #340,@#26 ;PRIO:7
2435 016456 104402 016462      TYPE    :+2 ;.ASCIZ <15><12>'POWER''
2436 016472 000137 007146      JMP     1$JMP ;JMP TO USER ADDRESS
2437
2438 016476 000000      .!LLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
2439 016500 000776      BR      -2 ; BEFORE THE POWER DOWN WAS COMPLETE
2440
2441 016502 000000      .SAVR6: 0 ;PUT THE SP HERE
2442 016504 000024 000026      .PUVEC: 24,26 ;POWER UP VECTOR

```

```

2443          .SBTTL          $TYPEA - 18 BIT ADDRESS TYPER
2444
2445          ;THIS ROUTINE TAKES 2 ARGUMENTS OFF THE STACK (OLD
2446          ;SP AND ADDRESS) AND, USING THE MEMORY MANAGEMENT REGISTERS, TYPES
2447          ;THE ADDRESS SUPPLIED IN 18 BIT FORM. THIS ROUTINE IS LINKED
2448          ;VIA THE 'TYPADR' MACRO.
2449
2450          .TYPEA:
2451          016510          MOV          R4,-(6)          ;PUSH R4 ON STACK
2452          016510          MOV          R5,-(6)          ;PUSH R5 ON STACK
2453          016514          016605          0C0012          MOV          12(6),R5          ;R5 - OLD PS WITH PREVIOUS MODE
2454          016520          016604          000010          MOV          10(6),R4          ;R4 - ADDRESS TO BE DECODED AND TYPED
2455          016524          016666          000006          000010          MOV          6(6),10(6)          ;MOVE
2456          016532          016666          000004          000006          MOV          4(6),6(6)          ;DOWN
2457          016540          016666          000002          000004          MOV          2(6),4(6)          ;FOUR
2458          016546          012616          MOV          (6)+,(6)          ;WORDS
2459          016550          010346          MOV          R3,-(6)          ;PUSH R3 ON STACK
2460          016552          000305          SWAB          R5          ;GET THE
2461          016554          006005          ROR          R5          ;2 PREVIOUS
2462          016556          006005          ROR          R5          ;MODE BITS
2463          016560          006005          ROR          R5          ;INTO POSITION
2464          016562          042705          177771          BIC          #177771,R5          ;TO USE AS AN OFFSET
2465          016566          016505          016742          MOV          .SATAB(5),R5          ;R5 - SPACE ADDRESS FOR MM
2466          016572          010403          MOV          R4,R3          ;R3 - REGISTER OFFSET
2467          016574          042704          160000          BIC          #160000,R4          ;CLEAR THE MM REG SELECT BITS
2468          016600          000303          SWAB          R3          ;NOW MAKE
2469          016602          006003          ROR          R3          ;MM REG
2470          016604          006003          ROR          R3          ;SELECT BITS
2471          016606          006003          ROR          R3          ;INTO AN
2472          016610          006003          ROR          R3          ;OFFSET
2473          016612          042703          177761          BIC          #177761,R3          ;CLEAR THE JUNK BITS
2474          016616          060305          ADD          R3,R5          ;ADD THE OFFSET TO THE TABLE
2475          016620          011505          MOV          (5),R5          ;GET THE ISAR DATA
2476          016622          005003          CLR          R3
2477          016624          006305          ASL          R5          ;THIS IS
2478          016626          006103          ROL          R3
2479          016630          006305          ASL          R5          ;TO SHIFT
2480          016632          006103          ROL          R3
2481          016634          006305          ASL          R5          ;THE SEGMENT
2482          016636          006103          ROL          R3
2483          016640          006305          ASL          R5          ;ADDRESS
2484          016642          006103          ROL          R3
2485          016644          006305          ASL          R5          ;AN 18 BIT
2486          016646          006103          ROL          R3          ;ADDRESS
2487          016650          006305          ASL          R5          ;POSITION
2488          016652          006103          ROL          R3          ;WITH R3 CONTAINING
2489          016654          060405          ADD          R4,R5          ;THE UPPER 2 BITS
2490          016656          005503          ADC          R3          ;AND R5 CONTAINING
2491          016660          006305          ASL          R5          ;THE 16 BIT ADDRESS
2492          016662          006103          ROL          R3          ;THEN SHIFT FOR TYPING
2493          016664          010346          MOV          R3,-(SP)
2494          016666          000241          CLC
2495          016670          006016          ROR          (SP)
2496          016672          000241          CLC
2497          016674          006016          ROR          (SP)
2498          016676          000241          CLC

```

2499	016700	006016		ROR	(SP)	
2500	016702	104406		TYPES		
2501	016704	042703	177770	BIC	#177770,R3	
2502	016710	110337	016326	MOVB	R3,.PR+2	:GET THE FIRST NUMBER FROM R3
2503	016714	062737	000060	ADD	#0,.PR+2	:MAKE IT INTO A NUMBER
2504	016722	012704	016327	MOV	#.PR+3,R4	:FUDGE IN THE POINTER
2505	016726	012737	175401	MOV	#175401,.PR	:AND THE FLAGS (FILL & 5 BYTES)
2506	016734	012603		MOV	(6)+,R3	:POP STACK INTO R3
2507	016736	000137	016210	JMP	.PRL	:DECODE AND TYPE THE REST
2508						
2509	016742	172340		.SATAB:	172340	:KISARO

```

2510 .SBTTL          $TRAP - TRAP HANDLER
2511
2512 ;THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE
2513 ;SUBROUTINE. THE CALL IS A 'TRAP+N' WHERE N IS A MULTIPLE OF 2.
2514 ;THE 'SET' MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO
2515 ;FOLLOW THIS MACRO.
2516
2517 016744 011646 .TRAP: MOV      (6),-(6)          ;GET ADDRESS OF TRAP +2
2518 016746 162716 SUB      #2,(6)          ;MAKE IT ADDRESS OF TRAP
2519 016752 017616 MOV      @ (6),(6)        ;GET TRAP INSTRUCTION
2520 016756 062716 ADD      #.TRP+2-TRAP,(6) ;GET DATA AND MAKE IT AN OFFSET
2521 016762 013607 .TRP:  MOV      @ (6)+,PC    ;GO TO PROPER SUBROUTINE
2522
2523 016764 015642 .SCOPE = TRAP+0      (104400)
2524 016766 015504 .TYPE  = TRAP+2      (104402)
2525 016770 016146 .TYPE0 = TRAP+4      (104404)
2526 016772 016156 .TYPE5 = TRAP+6      (104406)
2527 016774 020556 .TYPE6 = TRAP+10     (104410)
2528 016776 016510 .TYPEA = TRAP+12     (104412)
2529 017000 006212 .ERCLR = TRAP+14     (104414)
2530 017002 006234 .DKCMD = TRAP+16     (104416)
2531 017004 017024 .RDOCT = TRAP+20     (104420)
2532 017006 017142 .RDLIN = TRAP+22     (104422)
2533 017010 020524 .UPDAT = TRAP+24     (104424)
2534 017012 000342 .CLRDV = TRAP+26     (104426)
2535 017014 014014 .LOGW  = TRAP+30     (104430)
2536 017016 014024 .LOGR  = TRAP+32     (104432)
2537 017020 014032 .LOGWC = TRAP+34     (104434)
2538 017022 014044 .LOGC  = TRAP+36     (104436)
  
```

```

2539          .SBTTL          $RDOCT - OCTAL INPUT ROUTINE
2540
2541          ;THIS ROUTINE CALLS RDLIN, INPUTS A LINE FROM THE TTY AND CONVERTS
2542          ;IT INTO AN OCTAL NUMBER WHICH IS THE FIRST WORD ON THE STACK.
2543
2544 017024 011546          .RDOCT: MOV      (6),-(6)          ;MOVE THE PC
2545 017026 016666 000004 000002 MOV      4(6),2(6)        ;MOVE THE PS
2546 017034 010146          MOV      R1,-(6)          ;PUSH R1 ON STACK
2547 017036 010246          MOV      R2,-(6)          ;PUSH R2 ON STACK
2548 017040 010346          MOV      R3,-(6)          ;PUSH R3 ON STACK
2549 017042 104422          4$:  RDLIN          ;READ A LINE INTO INPUT
2550 017044 005001          CLR      R1          ;INIT DATA WORD
2551 017046 012703 017246          MOV      #INPUT,R3      ;INIT POINTER
2552 017052 112302          1$:  MOVB     (3)+,R2      ;GET A BYTE
2553 017054 001417          BEQ     2$          ;GET OUT IF ZERO
2554 017056 122702 000060          CMPB    #'0,R2        ;CHECK FOR 0 OR GREATER
2555 017062 003022          BGT     3$          ;ERROR - LESS THAN 0
2556 017064 122702 000067          CMPB    #'7,R2        ;CHECK FOR 7 OR LESS
2557 017070 002417          BLT     3$          ;ERROR - GREATER THAN 7
2558 017072 006002          ROR     R2          ;GET
2559 017074 006002          ROR     R2          ;INTO
2560 017076 006002          ROR     R2          ;POSITION
2561 017100 006101          ROL     R1          ;FIRST BIT
2562 017102 006102          ROL     R2          ;GET
2563 017104 006101          ROL     R1          ;SECOND BIT
2564 017106 006102          ROL     R2          ;GET
2565 017110 006101          ROL     R1          ;THIRD BIT
2566 017112 000757          BR      1$          ;LOOP
2567 017114 010166 000012          2$:  MOV      R1,12(6)      ;SAVE THE RESULT
2568 017120 012603          MOV      (6)+,R3      ;POP STACK INTO R3
2569 017122 012602          MOV      (6)+,R2      ;POP STACK INTO R2
2570 017124 012601          MOV      (6)+,R1      ;POP STACK INTO R1
2571 017126 000002          RTI          ;RETURN
2572
2573 017130          3$:
2574 017130 104402 017134          TYPE    2+2          ;.ASCIZ '??'<15><12>
2575 017140 000740          BR      4$          ;TRY AGAIN
  
```

2576  
2577  
2578  
2579  
2580  
2581  
2582  
2583 017142 010546  
2584 017144 012705 017246  
2585 017150 022705 017266  
2586 017154 001412  
2587 017156 105737 177560  
2588 017162 100375  
2589 017164 113715 177562  
2590 017170 142715 000200  
2591 017174 122715 000177  
2592 017200 001005  
2593 017202  
2594 017202 104402 017206  
2595 017212 000754  
2596 017214 111527 000000  
2597 017220 104402 017216  
2598 017224 122725 000015  
2599 017230 001347  
2600 017232 105065 177777  
2601 017236 104402 000012  
2602 017242 012605  
2603 017244 000002  
2604  
2605 017246 000020  
2606 017266 000020  
2607 017326 000000  
2608

```
.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.

.RDLIN: MOV      R5, -(6)          ;SAVE R5
1$:      MOV      #INPUT, R5      ;GET ADDRESS
2$:      CMP      #INPUT+16., R5   ;BUFFER FULL?
        BEQ      4$              ;YES - TYPE '?'
        TSTB     @#177560         ;WAIT FOR
        BPL      -4              ;A CHARACTER
        MOVB     @#177562, (5)    ;GET CHARACTER
        BICB     #200, (5)        ;GET RID OF JUNK
        CMPB     #177, (5)        ;IS IT A RUBOUT
        BNE      3$              ;SKIP IF NOT

4$:      TYPE     ..+2            ;.ASCIZ '?'<15><12>
        BR       1$              ;ZAP THE BUFFER AND LOOP
3$:      MOVB     (5), #0         ;SET UP FOR TYPING
        TYPE     .3$+2           ;ECHO IT
        CMPB     #15, (5)+       ;CHECK FOR RETURN
        BNE      2$              ;LOOP IF NOT RETURN
        CLRB     -1(5)           ;ZAP RETURN (THE 15)
        TYPE     .12             ;TYPE A LINE FEED
        MOV      (6)+, R5        ;RESTORE R5
        RTI                      ;RETURN

INPUT:   .BLKB  16.             ;TTY INPUT AREA
ERTAB:   .BLKW  16.
OUTBUF:  0
```

```
2609 :THIS SUBROUTINE HAS THE CALL
2610 :JSR PC,MMUSE
2611 :XXX
2612 :
2613 :WHERE XXX IS AN EXECUTABLE ONE WORD INSTRUCTION
2614 :MEM MANAGEMENT IS USED ON THE DESTINATION
2615 :THE DESTINATION FIELD MUST BE (R1)+
2616 :NO OTHER FORM IS ALLOWED
2617
2618
2619
2620 MMUSE: TST TEMP2
2621 BEQ 2$
2622 MOV (R5)+,1$
2623 BIC #160000,R1
2624 BIS #60000,R1
2625 $: .WORD 0 ;CONTAINS INSTRUCTION JUST AFTER
2626 ;THE JSR R5, MMUSE
2627 017354 032701 100000 BIT #100000,R1
2628 017360 001403 BEQ 2$
2629 017362 062737 000200 172346 ADD #200,KIPAR3
2630 017370 000205 2$: RTS R5
```

```
2631
2632
2633
2634
2635 :THIS SETUP MEM MANAGEMENT FOR A FOLLOWING INSTRUCTION
2636 :CALL JSR PC,MMPSET
2637
```

```
2638
2639 MMPSET: TST TEMP2
2640 BEQ 2$
2641 MOV TEMP3,-(6)
2642 CLR KIPAR3
2643 017410 062737 000200 172346 1$: ADD #200,KIPAR3
2644 017416 005316 DEC (6)
2645 017420 001373 BNE 1$
2646 017422 005726 TST (6)+
2647 017424 000207 2$: RTS PC
2648
2649
2650 TIEOUT: TYPE ,CRLF
2651 017432 104402 000757 TYPE ,+2 ;.ASCIZ "TIMEOUT PC="
2652 017452 162746 000002 SUB #2,-(SP)
2653 017456 104404 TYPEO
2654 017460 000000 HALT
2655
2656
2657
```

;NOTE FOR PROGRAMMER\*\*\*\*\* PROGRAM AT THIS POINT CAN NOT EXCEED A PC OF 17500\*\*\*\*\*

2658



```

2659          020000          = 20000
2660          :NOTE ALL THIS CODE GETS DESTROYED WHEN PATTERN IS WRITTEN
2661
2662          :ROUTINE TO SAVE ABS LOADER
2663 020000 012700 017776 LDR: MOV #17776,R0
2664 020004 012737 020024 000004 MOV #2$ ,4 ;SET TIME OUT TRAP VECTOR
2665 020012 012737 000340 000006 MOV #340,6
2666 020020 005720 TST (R0)+
2667 020022 000776 BR -2
2668 020024 022626 2$: CMP (SP)+,(SP)+
2669 020026 012737 000006 000004 MOV #6,4
2670 020034 005037 000006 CLR 6
2671 020040 162700 000334 SUB #334,R0 ;POINT R0 BACK TO LOADER
2672 020044 010037 015166 MOV R0,LDR1 ;SAVE FOR RESTORE ROUTINE
2673 020050 012702 000155 MOV #155,R2 ;WORD COUNT
2674 020054 012703 017500 MOV #17500,R3 ;WHERE LOADER IS TO BE STORED
2675 020060 012023 $: MOV (R0)+,(R3)+ ;STORE LOADER
2676 020062 005302 DEC R2
2677 020064 001375 BNE 1$
2678 020066 000207 RTS PC ;RETURN
2679
2680
2681          : -A- PORT SIZE
2682
2683 020070 052737 020000 001126 SIZZAP: BIS #BIT13,FLAG2 ;SET MAPPING BIT
2684 020076 004737 001620 JSR PC,DRVENO ;FIND DRIVE
2685 020102 012737 000002 001230 MOV #2,WORK1 ;START WITH ONE 4K BUFFER
2686 020110 012737 000001 001074 MOV #1,STAMEM ;FIRST 4K BANK
2687 020116 012737 057476 001144 MOV #57476,BUF ;GET STARTING ADDR. 5K
2688 020124 012737 000001 001134 MOV #1,WRDCT ;LOAD WC
2689 020132 005037 001134 CLR DMA ;LOAD DA
2690 020136 012777 000040 160670 MOV #40,@RSCS2 ;CLEAR ALL RS REG
2691 020144 013777 001164 160662 MOV UNUM,@RSCS2 ;GET DRIVE #
2692          :XYZ*****?*****
2693          :*****
2694          :*****
2695          :XYZ*****?*****
2696          :*****
2697          :*****
2698          :XYZ*****?*****
2699          :*****
2700 020152 005037 001026 CLR TEMP3
2701 020156 012737 000061 001176 MOV #61,CMD ;DO A ERITE
2702 020164 104416 4$: DKCMD ;NOW
2703 020166 105777 160640 1$: TSTB @RSCS1 ;DONE YET?
2704 020172 100375 BPL 1$ ;NO
2705 020174 032777 004000 160632 BIT #4000,@RSCS2 ;DID MEM SET?
2706 020202 001012 BNE SIZ1 ;YES
2707 020204 005777 160622 TST @RSCS1 ;ANY ERRORS?
2708 020210 100005 BPL 3$ ;NO
2709 020212 012737 000006 001112 MOV #6,SIZEAP ;GET SIZE OF BUFFER
2710 020220 001137 020326 JMP @#SIZERR ;FOR USER IF HE WISHES IT
2711 020224 104424 3$: UPDAT ;GET NEXT 4K BANK
2712 020226 000756 BR 4$ ;TEST BANK
2713 020230 005337 001230 SIZ1: DEC WCRK1 ;DEC SIZE OF BUFFER
2714 020234 013737 001230 001112 MOV WORK1,SIZEAP ;LOAD SIZE OF A BUFFER
  
```

```
2715 :XYZ*****?*****  
2716 :*****  
2717 :*****  
2718 :XYZ*****?*****  
2719 :*****  
2720 :*****  
2721 :XYZ*****?*****  
2722 :*****  
2723 020242 013737 001230 0G1022 MOV WORK1,TEMP1  
2724 020250 104402 020254 TYPE +2 ;.ASCIZ <15><12> 'PORT -A- DATA BUFFER 4K TO ''  
2725 020316 004737 020464 JSR PC,SIZPR
```

```

2726 020322 000137 020444      SIZZBP: JMP      NOB          ;GET OUT NO -B- PORT
2727 020326                      SIZERR:
2728 020326 104402 020332      TYPE      ..+2          ;.ASCIZ <15><12>'WILL NOT CONTINUE TO SIZE MEMORY BECAUS
2729 020416 012737 000006 001114  MOV      #6,SIZEBP      ;GIVE PROGRAM A BUFFER
2730 020424 104060          HLT      .BA!DS        ;YOU CAN ENTER CONVERSATION MODE
2731 020426 052737 000001 001126  BIS      #BIT0,FLAG2    ;BEEN HERE BEFORE FLAG
2732 020434 042737 020000 001126  BIC      #BIT13,FLAG2   ;CLEAR MAPPING FLAG
2733 020442 000000          HALT     ;OR GO TO DERSA
2734 020444 042737 020000 001126  NOB:    BIC      #BIT13,FLAG2 ;CLEAR MAPPING FLAG
2735 020452 052737 000002 001126  BIS      #BIT1,FLAG2    ;SET BEEN HERE FLAG
2736 020460 000137 001462          JMP      CALM          ;CAL BUFFER AND WC
2737
2738 020464 005001          SIZPR:  CLR      R1          ;INIT SETUP
2739 020466 012702 000004          MOV      #4,R2
2740 020472 062701 000001          SIZP:   ADD      #1,R1      ;SETUP FOR BANK NO
2741 020476 062702 000004          ADD      #4,R2          ;SETUP FOR SIZE FO MEMORY
2742 020502 020137 001230          CMP      R1,WORK1      ;IS THIS THE RIGHT SIZE?
2743 020506 001371          BNE     SIZP          ;NO
2744 020510 010246          MOV      R2,-(6)      ;PUT R2 ON STACK
2745 020512 104410          TYPED   ;TYPE STACK IN DECIMAL
2746 020514 104402 020520          TYPE    ..+2          ;.ASCIZ 'X'
2747 020522 000207          RTS     PC           ;RETURN
2748
2749                      ;ADD 4K TO TEST ADDR
2750
2751                      ;XYZ*****?*****
2752                      ;*****
2753                      ;*****
2754                      ;XYZ*****?*****
2755                      ;*****
2756                      ;*****
2757                      ;XYZ*****?*****
2758                      ;*****
2759 020524 005237 001230          .UPDAT: INC      WORK1      ;INC BANK #
2760 020530 062737 020000 001144          ADD      #20000,BUF    ;UPDATE BY 4K
2761 020536 103005          BCC     1$           ;BRANCH IF NO OVERFLOW
2762 020540 005237 001026          INC      TEMP3         ;INCREMENT FOR RSBAE
2763 020544 013777 001026 1603'0  MOV      TEMP3,RSBAE
2764 020552 000002          1$:     RTI
2765
2766 020554 020556          .SBTTL  .TYPED   ;TYPED = TRAP+40 (104440)
2767                      ;SBTTL  $TYPED - CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2768
2769 020556 012737 100040 021004  .TYPED: MOV      #100040,.DSIGN ;SET BLANK SWITCH AND SIGN
2770 020564 010046          MOV      R0,-(6)      ;PUSH R0 ON STACK
2771 020566 010146          MOV      R1,-(6)      ;PUSH R1 ON STACK
2772 020570 010246          MOV      R2,-(6)      ;PUSH R2 ON STACK
2773 020572 010346          MOV      R3,-(6)      ;PUSH R3 ON STACK
2774 020574 010546          MOV      R5,-(6)      ;PUSH R5 ON STACK
2775 020576 012737 100040 021004  MOV      #100040,.DSIGN ;SET BLANK SWITCH AND SIGN
2776 020604 016605 000016          MOV      16(6),R5     ;GET DATA TO BE TYPED
2777 020610 100004          BPL     1$           ;BR IF INPUT IS POS.
2778 020612 005405          NEG     R5           ;MAKE THE BINARY NUMBER POS.
2779 020614 112737 000055 021004  MOV/B   #'-.,.DSIGN   ;MAKE THE ASCII NUMBER NEG.
2780 020622 005000          1$:     CLR      R0          ;ZERO THE CONSTANTS INDEX
2781 020624 012703 020774          MOV      #.DBLK,R3    ;SETUP THE OUTPUT POINTER

```

2782	020630	112723	000040		MOV B	#' ,(R3)+	:SET THE FIRST CHARACTER TO A BLANK
2783	020634	005002		2\$:	CLR	R2	:CLEAR THE BCD NUMBER
2784	020636	016001	020764		MOV	.DTBL(R0),R1	:GET THE CONSTANT
2785	020642	160105		3\$:	SUB	R1,R5	:FORM THIS BCD DIGIT
2786	020644	002402			BLT	4\$	:BR IF DONE
2787	020646	005202			INC	R2	:INCREASE THE BCD DIGIT BY 1
2788	020650	000774			BR	3\$	
2789	020652	060105		4\$:	ADD	R1,R5	:ADD BACK THE CONSTANT
2790	020654	005702			TST	R2	:CHECK IF BCD DIGIT=0
2791	020656	001003			BNE	5\$	:FALL THROUGH IF 0
2792	020660	105737	021005		TSTB	.DSIGN+1	:STILL DOING LEADING 0'S?
2793	020664	100410			BMI	7\$	:BR IF YES
2794	020666	106337	021005		ASLB	.DSIGN+1	:MSD?
2795	020672	103003			BCC	6\$	:BR IF NO
2796	020674	113763	021004	*77777	MOV B	.DSIGN,-1(R3)	:YES--SET THE SIGN
2797	020702	052702	000060		BIS	#'0,R2	:MAKE THE BCD DIGIT ASCII
2798	020706	052702	000040		BIS	#' ,R2	:MAKE IT A SPACE IF NOT ALREADY A DIGIT
2799	020712	110223			MOV B	R2,(R3)+	:PUT THIS CHARACTER IN THE OUTPUT BUFFER
2800	020714	005720			TST	(R0)+	:JUST INCREMENTING
2801	020716	020027	000010		CMP	R0,#10	:CHECK THE TABLE INDEX
2802	020722	002744			BLT	2\$	:GO DO THE NEXT DIGIT
2803	020724	003002			BGT	8\$	:GO TO EXIT
2804	020726	010502			MOV	R5,R2	:GET THE LSD
2805	020730	000764			BR	6\$	:GO CHANGE TO ASCII
2806	020732	105013		8\$:	CLRB	(R3)	:SET THE TERMINATOR
2807	020734	012605			MOV	(6)+,R5	:POP STACK INTO R5
2808	020736	012603			MOV	(6)+,R3	:POP STACK INTO R3
2809	020740	012602			MOV	(6)+,R2	:POP STACK INTO R2
2810	020742	012601			MOV	(6)+,R1	:POP STACK INTO R1
2811	020744	012600			MOV	(6)+,R0	:POP STACK INTO R0
2812	020746	016666	000002	000004	MOV	2(6),4(6)	:FUDGE DATA
2813	020754	012616			MOV	(6)+,(6)	:OFF STACK
2814	020756	104402	020774		TYPE	..DBLK	:NOW TYPE THE NUMBER
2815	020762	000002			RTI		:RETURN
2816							
2817	020764	023420	001750	000144	.DTBL:	10000.,1000.,100.,10.	
2818	020772	000012					
2819	020774	000004			.DBLK:	.BLKW 4	
2820	021004	000000			.DSIGN:	0	
2821							
2822		000001			.END		



CMPLP1	010610	1517	1519	1531#															
CMFY	003234	496	520	538	567	573	579	591#											
COMDAR	007256	1256	1258#																
COMERR	001214	338#	2014	2034*	2171*	2172													
COMPAR	010414	819	1013	1491#															
CONM	002220	405	476#																
CRLFLF	000757	199#	447	673	1001	1716	2178	2222	2650										
DA	- 000004	246#	639	653	729	787	833	957	989	1029	1513	1795	1847	1863					
		1971	2120																
DATA	000620	174#	727	765	830														
DATAT	003660	589	685#	883															
DATTES	002402	497	500#	505															
DB	000002	245#	787	989	1029	1847	1863	1919	1971	2135									
DISBUF	007204	641	678	740	799	844	1247#	1850	1867	1923	1973								
DISPLA=	177570	127#	1345*	2315*	2319*														
DKCMD	- 104416	631	646	716	750	813	943	971	1009	1843	1858	1915	1964	1967					
		2530#	2702																
DKINT	006422	1119#	1227																
DMA	001140	316#	392*	615*	627	561	667	701*	929*	930*	1055*	1077	1153*	1252*					
		1253	1261	1266	1277*	1291	1616	1839*	1854*	1910*	1955*	2689*							
DONE	013752	2022	2027	2039#															
DONEE	002204	460	470#	1062															
DOWN	012624	1840	1878#	1881															
DROP	001130	274#	418	1810*	1981	2018	2174*	2179											
DRP	014760	2074	2163	2165#															
DRVEND	001620	409#	585	2684															
DS	= 000040	249#	729	833	957	1129	1513	1795	2130	2730									
DSKRD	004540	809	811#	837															
DT	= 000240	253#	2142																
DVNUM	001660	414	417#																
ELH	004604	820	822#																
ERCLR	= 104414	624	644	711	743	811	941	969	1007	2529#									
ERCOUN	001152	321#	1491*	1626*															
ERRCL	012300	370	493	1803#															
ERRORS	001002	209#	2334*	2354															
ERRVEC=	000004	349#	402*	403*															
ERTAB	017266	1804	2030	2071*	2606#														
ESH	004474	802#	893																
ESH1	004470	747	801#																
EXGEN	010200	1421	1437#																
EXGEN1	010336	1472#																	
EXMFLG	003212	584	586#	1061	2247														
EXRAX	006076	963	995	1017	1035#														
EXRXX	006110	1036	1038#																
EXTDR	007546	1283	1289	1312#															
EXTMEM	011464	386	490	1673#															
EXTPP	006206	1059	1062#																
EXTPPR	006154	890	1055#	2188															
EXTT	011442	1656	1661#																
FILBUF	007764	1365	1367#																
FILCHR	001014	213#	2268	2269															
FILDAT	007760	1366#	1369																
FLAG	001132	313#	362*	397*	400*	489*	492*	494*	498*	540*	560*	569*	575*	581*					
		583	586*	587	613*	637*	651*	681*	708*	713	719	730*	731	739*					
		745	752	803	807*	917	822	846	887	913*	931*	945	959*	960					
		973	1003*	1016	1043*	1058	1067	1070*	1103	1119*	1139*	1151	1247	1258					

		1275	1279*	1280	1284	1311*	1350	1629	163*	1845	1861	1917	1953*	1969
FLAG2	001126	2185												
		273#	363*	365	383	388	391*	410*	424*	438	445	452*	459	471*
		516*	1107	1110	1143	1185*	1187*	1501*	1568*	1650*	1730*	1732*	1733*	1736*
		1769*	1797*	1864*	1869	1872*	1903*	1920*	1925	1927	1930*	1931*	1941*	1954*
		2079*	2157	2221*	2232	2683*	2731*	2732*	2734*	2735*				
FLAG3	001222	341#	364*	595*	682	894*	1044							
FNDTYP	006740	507	1185#	1836										
GETERR	013720	1996	2028#	2168										
HERADD=	177742	351#	2224											
HINUM	010206	1376	1387	1397	1422*	1439#								
HINUM1	010342	1450	1460	1471*	1474#	1506*	1529							
HISAV	010212	1376*	1441#	1506										
HLT =	104000	123#	639	653	669	729	787	833	957	989	1029	1129	1268	1513
		1587	1795	1847	1863	1919	1971	2730						
HLTADR	001012	212#	2338*	2339*	2340	2341	2355							
HRDR	001156	323#	706*	922*	1137	1853*	1904*							
ICNT	001000	208#	360*	2308	2310	2312*	2313*	2315	2318*	2319	2325	2350*		
INCSEC	007232	1252#	1257											
INPUT	017246	593	2551	2584	2585	2605#	.							
INTEXT	006512	1122	1140#											
INTFLG	001202	333#	1098*	1146*	1515	1520	1570							
KDPAR0=	172360	113#												
KDPAR1=	172362	114#												
KDPAR2=	172364	115#												
KDPAR3=	172366	116#												
KDPAR4=	172370	117#												
KDPAR5=	172372	118#												
KDPAR6=	172374	119#												
KDPAR7=	172376	120#												
KDPDR0=	172320	95#												
KDPDR1=	172322	96#												
KDPDR2=	172324	97#												
KDPDR3=	172326	98#												
KDPDR4=	172330	99#												
KDPDR5=	172332	100#												
KDPDR6=	172334	101#												
KDPDR7=	172336	102#												
KIPAR0=	172340	104#	1722*											
KIPAR1=	172342	105#	1724*											
KIPAR2=	172344	106#	1740*											
KIPAR3=	172346	107#	1564*	1742*	2629*	2642*	2643*							
KIPAR4=	172350	108#	1744*											
KIPAR5=	172352	109#	1746*											
KIPAR6=	172354	110#	1748*											
KIPAR7=	172356	111#	1757*											
KIPDR0=	172300	86#	1749*											
KIPDR1=	172302	87#	1750*											
KIPDR2=	172304	88#	1751*											
KIPDR3=	172306	89#	1752*											
KIPDR4=	172310	90#	1753*											
KIPDR5=	172312	91#	1754*											
KIPDR6=	172314	92#	1755*											
KIPDR7=	172316	93#	1756*											
LA	000204	251#	2152											
LAD	001010	211#	361*	636*	718*	798*	925*	2314*	2320	2322	2325			









SR3 = 172516	84#	1760*																	
STABUF 006566	476	480	1155#																
STAMEM 001074	260#	478*	699	1201	1673	1688	2686*												
STATUS 001070	255#	705*	924*																
STBCOM 001100	262#	483*																	
STMM2 011650	1658	1719#																	
STTEST 001752	422	425#	457																
SWITCH 001200	332#	1504*	1522	1526*	1528*														
SWR - 177570	126#	127	373	395	398	421	443	757	759	770	815	824	885						
	947	975	1011	1019	1021	1140	1482	1572	1581	1624	1704	1835	1977						
	2021	2077	2162	2165	2243	2295	2306	2335	2345	2348									
SWRDCT 001150	320#	390	394	401	533*	534	614	618*	622*	680*	700	1282	1443*						
	1444	1693*	1958*	1959															
SW0 = 000001	22#																		
SW1 - 000002	21#																		
SW10 = 002000	12#	2042	2334																
SW11 = 004000	11#	2306																	
SW12 = 010000	10#	2042																	
SW13 = 020000	9#	2335																	
SW14 = 040000	8#	2295																	
SW15 = 100000	7#																		
SW2 = 000004	20#																		
SW3 = 000010	19#																		
SW4 = 000020	18#																		
SW5 = 000040	17#																		
SW6 - 000100	16#																		
SW7 = 000200	15#																		
SW8 000400	14#	2295																	
SW9 - 001000	13#	2348																	
TAC 004604	816	818	821#																
TAG1 005774	1012	1015#																	
TDMA 001146	319#	550*	1153																
TEMP1 001022	217#	865	868	2723*															
TEMP2 001024	218#	611*	698*	912*	1056*	1086	1209	1364	1416	1431	1541	1558	1644						
	1831*	1902*	2620	2639															
TEMP3 001026	219#	699*	864*	865	867*	868	870*	872	1088	1211	2641	2700*	2762*						
	2763																		
TEMP4 001030	220#	1090*	1091*	1092															
TIEOUT 017426	402	2650#																	
TIMES 016012	2310	2325#																	
TIMUP 007146	1235#	2436																	
TPB 001020	215#	2269*	2274*																
TPS 001016	214#	2270	2275																
TRACK 001136	315#																		
TRUERR 006504	1126	1128	1139#																
TRY 001640	412#	416																	
TRVNX 002130	426	436	442	454#	1060														
TST1 003260	609#																		
TST2 003660	689#																		
TST3 005136	903#																		
TST4 006152	1054#																		
TYPE = 104402	423	447	451	465	476	480	481	484	495	501	519	524	537						
	541	552	566	572	578	591	673	727	728	765	766	780	784						
	830	831	955	956	982	983	986	993	998	1001	1027	1028	1181						
	1272	1479	1512	1586	1590	1620	1623	1666	1713	1716	1764	1794	1832						
	1871	1929	1975	1993	1998	1999	2000	2003	2004	2005	2008	2009	2010						





.TYPED	020556	2527	2766	2769#
.TYPEJ	016146	2363#	2525	
.TYPES	016156	2365#	2526	
.UPDAT	020524	2533	2759#	

END	1#														
LDPDR	1718#	1749	1750	1751	1752	1753	1754	1755	1756						
NEWST	1#	606	686	900	1051										
ODT11	1#														
ODT11X	1#														
POP	1#	153#	2301	2304	2396	2427	2506	2568	2807						
PRINT	1#	153#	423	481	495	501	519	524	537	541	552	566	572	578	780
	784	986	1272	1479	1586	1590	1620	1623	1871	1929	1975	1993	1998	2000	2003
	2005	2008	2010	2013	2016	2023	2026	2082	2085	2089	2093	2097	2100	2101	2104
	2107	2112	2117	2122	2127	2132	2137	2144	2149	2154	2159	2164	2175	2220	2223
	2226	2229	2234	2337	2343	2435	2573	2593	2651	2724	2728	2746			
PUSH	1#	153#	2298	2367	2413	2450	2459	2546	2770						
SCOPE.	1#	606#	686#	900#	1051#										
SET	2510#	2523	2524	2525	2526	2527	2528	2529	2530	2531	2532	2533	2534	2535	2536
	2537	2538	2766												
TYPADR	1#														
TYPBIT	1#	153#													
TYPCHR	1#	153#													
TYPDEC	1#	153#	2744												
TYPLIN	1#	153#													
TYPOCS	1#	153#	448	463	1273	1621	1994	2001	2006	2011	2014	2024	2160	2176	
TYPOCT	1#	153#	781	785	987	2083	2086	2090	2094	2098	2102	2105	2113	2118	2123
	2128	2133	2138	2145	2150	2155	2341								
TYPTXT	1#	153#	423	481	495	500	518	523	536	541	551	565	571	577	779
	784	986	1272	1478	1586	1590	1620	1623	1871	1929	1975	1992	1998	2000	2003
	2005	2008	2010	2013	2016	2023	2026	2082	2085	2088	2092	2096	2100	2101	2104
	2107	2112	2117	2122	2127	2132	2137	2144	2149	2154	2159	2164	2175	2219	2223
	2226	2229	2234	2651	2724	2727	2746								
SCATCH	1#	23													
SCMTAG	1#	206													
\$DONE	1#	2037													
\$EQUAT	1#	122													
\$HLT	1#	2326													
\$KMPR	1#	79													
\$OCTAL	1#	2355													
\$POWER	1#	2403													
\$RAND2	1#														
\$RAND4	1#														
\$RDDEC	1#														
\$RDLIN	1#	2576													
\$RDOCT	1#	2539													
\$SCOPE	1#	2288													
\$SET	2510#	2523	2524	2525	2526	2527	2528	2529	2530	2531	2532	2533	2534	2535	2536
	2537	2538	2766												
\$SETUP	1#	353													
\$SMMPR	1#														
\$SWDOC	1#	4													
\$TRAP	1#	2510													
\$TYPE	1#	2248													
\$TYPEA	1#	2443													
\$TYPED	1#	2767													
\$UMPR	1#														

. ABS. 021006 000

CERSB-C RH70-RS03 DATA AND RELIABILITY TEST      MACY11 30A(1052)      F 8      18-AUG-78 08:26      PAGE 85  
CERSBC.P11      14-AUG-78 08:29      CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0096

ERRORS DETECTED: 0

CERSBC.BIN,CERSBC.LST/CRF/SOL/NL:TOC=DSKZ:CERSBC.SML,DSKZ:CERSBC.P11  
RUN-TIME: 9 11 1 SECONDS  
RUN-TIME RATIO: 58/20=2.8  
CORE USED: 21K (41 PAGES)