

RH11,RS03

DATA RELIABILITY
CZRSCGO

AH-9319G-MC

APR 1978

COPYRIGHT ©73-78

digital

FICHE 1 OF 1

MADE IN USA

1	1	1	1	1	1	1
2	2	2	2	2	2	2
3	3	3	3	3	3	3
4	4	4	4	4	4	4
5	5	5	5	5	5	5
6	6	6	6	6	6	6
7	7	7	7	7	7	7
8	8	8	8	8	8	8
9	9	9	9	9	9	9
10	10	10	10	10	10	10
11	11	11	11	11	11	11
12	12	12	12	12	12	12
13	13	13	13	13	13	13
14	14	14	14	14	14	14
15	15	15	15	15	15	15
16	16	16	16	16	16	16
17	17	17	17	17	17	17
18	18	18	18	18	18	18
19	19	19	19	19	19	19
20	20	20	20	20	20	20
21	21	21	21	21	21	21
22	22	22	22	22	22	22
23	23	23	23	23	23	23
24	24	24	24	24	24	24
25	25	25	25	25	25	25
26	26	26	26	26	26	26
27	27	27	27	27	27	27
28	28	28	28	28	28	28
29	29	29	29	29	29	29
30	30	30	30	30	30	30

11

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 3
DESCRIPTION

1. ABSTRACT

THIS DIAGNOSTIC WAS DESIGNED TO TEST R503, R503/LA AND R504 DRIVES.

THE DZRSC DISK DATA TEST IS A SERIES OF ADDRESS AND DATA RELIABILITY ROUTINES WHICH VERIFY TO THE USER THAT THE CONTROLLER (RH11) AND THE DISKS (R503/LA OR R504) ARE OPERATING CORRECTLY. THIS TEST SHOULD BE USED IN CONJUNCTION WITH THE DZRSB DIAGNOSTIC. IF THERE IS A POWER FAIL WHILE THE DIAGNOSTIC IS RUNNING, THE PROGRAM WILL WAIT FOR APPROXIMATELY 5 MINUTES, TO GIVE ALL THE DRIVES TIME TO COME BACK UP TO SPEED, BEFORE RESTARTING THE TEST SEQUENCE.

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 RH11 CONTROLLER WITH AN R503, R503/LA OR AN R504 DISK.

2.2 PRELIMINARY PROGRAMS

DZRSB

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 4
DESCRIPTION

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5. (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.). ALL DOWN FOR WORST CASE (IF SWITCH-LESS CPV SIMPLY PRESS START)

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- AND -B- PORTS UP TO 124K. 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.

THE PROGRAM WILL MAKE DATA BUFFERS IN 4K SEGMENTS NOT TO EXCEED 24K. THE OPERATOR CAN SPECIFY THE BEGINNING BANK AND THE SIZE OF THE DATA BUFFER AREA (UP TO 24K) BY ENTERING THE CONVERSATION MODE. OR, BY DEFAULT, THE DATA BUFFERS WILL BEGIN AT THE FIRST AVAILABLE BANK AND USE AVAILABLE MEMORY UP TO 24K SIZE.

143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182

AC-9318G-MC CZRSCG RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 5
DESCRIPTION

5. OPERATIONAL SWITCH SETTINGS

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

(I.E) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

- 1. <CR> IF NO CHANGES ARE TO BE MADE.
- 2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER
VALUE: LAST DIGIT FOLLOWED BY <CR>.
- 3. ↑U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED
KEYING IN SWREG VALUE.

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

11003
11004
11005
11006
11007
11008
11009
11010
11011
11012
11013
11014
11015
11016
11017
11018
11019
11020
11021
11022
11023
11024
11025
11026
11027
11028
11029
11030
11031
11032
11033
11034
11035
11036
11037
11038
11039
11040
11041
11042
11043
11044
11045
11046
11047
11048
11049
11050
11051
11052
11053
11054
11055
11056
11057
11058
11059
11060
11061
11062
11063
11064
11065
11066
11067
11068
11069
11070
11071
11072
11073
11074
11075
11076
11077
11078
11079
11080
11081
11082
11083
11084
11085
11086
11087
11088
11089
11090
11091
11092
11093
11094
11095
11096
11097
11098
11099
11100

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 6
DESCRIPTION

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<08> = 1 DATA RELIABILITY TEST MODE
- 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 TYPEOUT OF RETRY ERRORS
- SW<05> = 1 INHIBIT PASS COUNT
- SW<04> = 1 ALLOWS 8 ERROR TYPEOUTS IN THE COMPARE ROUTINE BEFORE EXECUTING NEXT READ COMMAND. WHEN SWITCH IS 0, ONLY 1 ERROR TYPEOUT IS RECORDED.
- SW<03> = 1 TYPEOUT # OF ERRORS
- SW<02> = 1 INHIBIT MEMORY MANAGEMENT
- SW<01> = 1 DATA TEST ONLY
- SW<00> = 1 DROPS DRIVE AFTER 20 ERRORS

15
14
13
12
11
10
09
08
07
06
05
04
03
02
01
00

AC-9318G-MC CZRSCG RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 7
DESCRIPTION

5.1 DATA RELIABILITY TEST MODE

WITH SWB 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 OVER RIDES 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 SWB 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.

248
249
250
251
252
253
254
255
256
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

J01

AC-9318G-MC CZRSCG RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 9
DESCRIPTION

3. 1ST 4K BANK #

NOTE

THIS DIAGNOSTIC WILL ONLY TEST -B- PORT
IF THE PROCESSOR HAS ACCESS TO THAT
MEMORY ON -B- PORT. THIS MEMORY MUST
HAVE THE SAME ADDRESS ON ALL PORTS.

THIS NUMBER WILL DETERMINE WHERE THE DATA BUFFER AREA WILL
START ON -B- PORT.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

AC-9318G-MC CZRSCG RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 10
DESCRIPTION

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.

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. IF THE WORD COUNT IS NOT SPECIFIED, THE PROGRAM WILL USE AVAILABLE MEMORY UP TO 24K TO CREATE THE DATA BUFFERS.

WD CT

THE OPERATOR CAN SPECIFY ANY LENGTH TRANSFER FROM 1(8) TO 6000(8) WORDS. THE NORMAL TRANSFER LENGTH IS N(8) WORDS WHERE N IS THE MAXIMUM BUFFER SIZE FOR THE AVAILABLE CORE. IN EITHER CASE, BUFFER WILL NOT EXCEED 24 K.

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

348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402

AC-931BG-MC CZRSCG RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 11
DESCRIPTION

PATTERN NO. ?

THIS GIVES THE OPERATOR THE OPTION OF SELECTING ALL THE DATA PATTERNS (#22) OR ANY ONE DATA PATTERN, SIMPLY BY TYPING THE DATA PATTERN NUMBER DESIRED.

- 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

403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 12
DESCRIPTION

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. ONE SECTOR OF RANDOM DATA IS 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. IF THERE ARE DATA BUFFERS ON -A- AND -B- PORTS, THE DATA TEST WILL TRANSFER DATA OVER -A- PORT ON ODD PASSES AND OVER -B- PORT ON EVEN PASSES.

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.

452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502

503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 13
DESCRIPTION

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 SWD 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

THIS DIAGNOSTIC WILL TEST -B- PORT, ONLY IF THE CPU CAN ACCESS THAT MEMORY ON -B- PORT.

8. MISCELLANEOUS

8.1 EXECUTION TIME

PASS COMPLETE WILL BE TYPED OUT AT END OF PASS. IT WILL TAKE 10 TO 20 MINUTES TO COMPLETE A PASS DEPENDING ON THE TYPE OF DRIVE BEING TESTED AND THE SIZE OF THE SYSTEM.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 500

AC-9318G-MC CZRSCG RH11-R503/LA-R504 DATA RELIABILITY DIAG PAGE 14
DESCRIPTION

B.3 POWER FAIL

THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 244. WHEN ASKED, ENTER UNIT #. 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 250. 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 CZRSCG RH11-R503-R503/LA-R504 DATA AND RELIABILITY TEST
:COPYRIGHT 1973,1974,1975,1976,1977,1978 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
:PROGRAM BY STANLEY HARACKIEWICZ

575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001
000000
000046
014110
000052
040000
000174
000000
000176
000000
000200
000137
001230
000210
012706
000500
000214
000137
003232

```

:          SWITCH          USE
:          -----          -----
:          SW15= 100000    ;HALT ON ERROR
:          SW14= 40000     ;LOOP ON FUNCTION
:          SW13= 20000     ;INHIBIT ERROR TYPEOUTS
:          SW12= 10000     ;INHIBIT COMPARISON
:          SW11= 4000      ;HALT ON COMPLETION OF TRANSFER
:          SW10= 2000      ;CONVERSATION MODE
:          SW9= 1000       ;LOOP ON ERROR
:          SW8= 400        ;DATA RELIABILITY TEST MODE
:          SW7= 200        ;WAIT IN BACKGROUND TEST
:          SW6= 100        ;OPTIONAL TYPEOUT OF RETRY ERRORS
:          SW5= 40         ;INHIBIT PASS COUNT AND UNIT #
:          SW4= 20         ;ALLOWS 8 LOCATIONS TO BE TESTED IN COMPARE ROUTINE
:          SW3= 10         ;TYPE OUT TOTAL # OF ERRORS
:          SW2= 4          ;INHIBIT MEMORY MANAGEMENT
:          SW1= 2          ;DATA TEST ONLY
:          SW0= 1          ;DROP DRIVE AFTER 20 ERRORS
:          = 0             ;TRAP CATCHER FROM 0 - 776
:          SENDAD 46       ;HOOKS FOR ACT 11
:          BIT14 52
:          = 174          ;SOFTWARE SWITCH REGISTER LOCATION
:          DISPREG: 0
:          SWREG: 0
:
:NOTE: FOR PROGRAM TO BE RESTARTED AT 200 IT MUST BE RELOADED
:      INTO MEMORY DUE TO ONCE ONLY CODE.
:
:          = 200          ;
:          JMP          SETSWI      ;START TEST
:
:          = 210          ;
:          MOV          #500,SP      ;SETUP STACK
:          JMP          @#ADTST      ;RESTART ADDR

```

612							
613		000220		. =	220		
614	000220	012706	000500		MOV	#500.SP	: CONVERSATION MODE WITHOUT
615	000224	000137	002406		JMP	@#A1	: DATA BUFFER QUESTIONS
616							
617	000230	000137	015174		JMP	@#RLDR	: RESTORE LOADER
618							
619	000234	000137	003324		JMP	@#ADTL	: TRACK AND SECTOR SELECT TEST
620							: WRITE EACH WORD ADDR ON ITSELF AND READ IT BACK
621	000240	000137	005054		JMP	@#RANEL	: RANDOM ADDRESS, DATA TEST
622	000244	000137	012506		JMP	@#PFT1	: DISK WRITE POWER FAIL TEST
623	000250	000137	013034		JMP	@#PFT2	: DISK WRITE CHECK POWER FAIL TEST

624 ;RH11 DATA PATTERNS

625						PAT0:	0
626	000254	000000				PAT1:	177777
627	000256	177777				PAT2:	031463
628	000260	031463				PAT3:	066666
629	000262	066666				PAT4:	100001
630	000264	100001				PAT5:	107070
631	000266	107070				PAT6:	070707
632	000270	070707				PAT7:	052525
633	000272	052525				PAT10:	125252
634	000274	125252				PAT11:	177737
635	000276	177737				PAT12:	146314
636	000300	146314				PAT13:	136363
637	000302	136363				PAT14:	063636
638	000304	063636				PAT15:	000001
639	000306	000001				PAT16:	100005
640	000310	100005				PAT17:	155555
641	000312	155555				PAT20:	133333
642	000314	133333				;PAT21	RANDOM DATA

643											
644											
645											
646	000316	012777	000040	000510		;CLEAR ALL REGISTERS					
647	000324	013777	001160	000502		.CLRQV: MOV #40,QRSCS2 ;CLEAR ALL REG					
648	000332	000002				MOV UNNUM,QRSCS2 ;GET UNIT #					
649						RTI					

```

        .SBTTL          $KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS
650
651
652          177572          SR0=177572          ;ADDRESS OF MEM MGMT REGISTER SR0
653          177574          SR1=177574          ;          "          "          "          "          SR1
654          177576          SR2=177576          ;          "          "          "          "          SR2
655          172516          SR3=172516          ;ADDRESS OF MEM MGMT REGISTER SR3
656
657          172300          KIPDR0=172300          ;ADDRESS OF KERNEL 'I' PAGE
658          172302          KIPDR1=172302          ;DESCRIPTOR REGISTERS
659          172304          KIPDR2=172304
660          172306          KIPDR3=172306
661          172310          KIPDR4=172310
662          172312          KIPDR5=172312
663          172314          KIPDR6=172314
664          172316          KIPDR7=172316
665
666          172320          KDPDR0=172320          ;ADDRESSES OF KERNEL 'D' PAGE
667          172322          KDPDR1=172322          ;DESCRIPTOR REGISTERS
668          172324          KDPDR2=172324
669          172326          KDPDR3=172326
670          172330          KDPDR4=172330
671          172332          KDPDR5=172332
672          172334          KDPDR6=172334
673          172336          KDPDR7=172336
674
675          172340          KIPAR0=172340          ;ADDRESSES OF KERNEL 'I' PAGE
676          172342          KIPAR1=172342          ;ADRESS REGISTERS
677          172344          KIPAR2=172344
678          172346          KIPAR3=172346
679          172350          KIPAR4=172350
680          172352          KIPAR5=172352
681          172354          KIPAR6=172354
682          172356          KIPAR7=172356
683
684          172360          KDPAR0=172360          ;ADDRESSES OF KERNEL 'D' PAGE
685          172362          KDPAR1=172362          ;ADDRESS REGISTERS
686          172364          KDPAR2=172364
687          172366          KDPAR3=172366
688          172370          KDPAR4=172370
689          172372          KDPAR5=172372
690          172374          KDPAR6=172374
691          172376          KDPAR7=172376

```

F02

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 CZRSCG.F11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 19
 \$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0018

692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723

000001
104000
177776
177776
000007
000000
000001
000002
000003
000004
000005
000006
000007
000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000
000001
000000

GOOD=
BAD=

N= 1
 HLT= EMT
 PS= 177776
 PSW= PS
 BELL= 7
 R0= %0
 R1= %1
 R2= %2
 R3= %3
 R4= %4
 R5= %5
 SP= %6
 PC= %7
 BIT0= 1
 BIT1= 2
 BIT2= 4
 BIT3= 10
 BIT4= 20
 BIT5= 40
 BIT6= 100
 BIT7= 200
 BIT8= 400
 BIT9= 1000
 BIT10= 2000
 BIT11= 4000
 BIT12= 10000
 BIT13= 20000
 BIT14= 40000
 BIT15= 100000
 R1
 R0

```

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

:FOR GOOD DATA
:FOR BAD DATA
  
```


H02

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 21
\$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEG 0020

```
772          001000          . =      1000
773
774 001000 000000          ICNT:  0          ;LH = ITERATION COUNT ;RH = TEST NO.
775 001002 000000          ERRORS: 0          ;ERROR COUNT
776 001004 000000 000000 PCNT:  0,0        ;2 WORD PASS COUNT
777 001010 C00000          LAD:  0          ;LOOP ADDRESS FOR SCOPE
778 001012 000000          HLTADR: 0          ;ADDRESS OF LAST HLT INSTRUCTION EXECUTED
779 001014 001000          FILCHR: 1000       ;FILCHR=0 (CHAR) ;FILCHR+1=2 (COUNT)
780 001016 177564          TPS:  177564       ;OUTPUT STATUS REGISTER
781 001020 177560          TKS:  177560
782 001022 177562          TKB:  177562
783 001024 177566          TPB:  177566
784 001026 177570          SWR:  177570
785 001030 177570          DISPLAY: 177570
786
787          ;DISK I/O REGISTERS
788
789 001032 172040          RSCS1:  172040       ;DISK CONTROL + STATUS REGISTER
790 001034 172050          RSCS2:  172050       ;DISK CONTROL + STATUS REGISTER
791 001036 172042          RSWC:  172042       ;WORD COUNT REGISTER
792 001040 172044          RSBA:  172044       ;BUS ADDRESS
793 001042 172046          RSDA:  172046       ;DISK ADDRESS (DESIRED ADDRESS)
794 001044 172052          RSDS:  172052       ;DRIVE STATUS
795 001046 172054          RSER:  172054       ;ERROR REG.
796 001050 172056          RSAS:  172056       ;ATTENTION SUMMARY
797 001052 172060          RSLA:  172060       ;LOOK AHEAD
798 001054 172062          RSDB:  172062       ;DATA BUFFER REGISTER
799 001056 172064          RSMR:  172064       ;MAINTENANCE REGISTER
800 001060 172066          RSDT:  172066       ;DRIVE TYPE REGISTER
801 001062 000204          RSVEC:  204         ;INTERUPT RSVEC
802
803          ;BIT ASSIGNMENTS FOR ERROR TYPE OUTS
804
805          000002          DB=2          ;DATA BUFFER
806          000004          DA=4          ;DESIRED ADD
807          000010          WC=10         ;WORD COUNT
808          000020          BA=20         ;BUS ADDRESS
809          000040          DS=40         ;DRIVE STATUS
810          000100          AS=100        ;ATTENTION SUMMARY
811          000204          LA=204        ;LOOK AHEAD
812          000220          MR=220        ;MAINTENANCE
813          000240          DT=240        ;DRIVE TYPE
814
815 001064 000206          STATUS: 206       ;DISK INTERRUPT STATUS
816 001066 000200          PRIORITY:BIT7     ;DISK PRIORITY LEVEL
```

017		000006	RW=6	:R/W IN PDR REG
018		000000	UP=0	:UP BITY IN PDR REG
019		000250	MMVEC=250	:ADDR OF MEM MGMT ERROR TRAP
020	001070	000000	STAMEM: 0	:STARTING LOC FOR -A- PORT
021	001072	000000	SAVAST: 0	:SAVE LOC FOR STAMEM
022	001074	000000	STBCOM: 0	:STARTING LOC FOR -B- PORT
023	001076	000000	SAVCPU: 0	:SAVE LOC FOR CPUBM
024	001100	000000	SAVMGA: 0	:STARTING ADDR FOR -A- PORT WITH MEM MGMT
025	001102	000000	SAVMGB: 0	:STARTING ADDR FOR B PORT W/MEM MGMT
026	001104	000000	SAVMGC: 0	:STARTING LOC FOR CPU W/MEM MGMT
027	001106	000000	SIZEAP: 0	:SIZE OF A PORT
028	001110	000000	SIZEBP: 0	:SIZE OF B PORT
029	001112	000000	WDCTB: 0	:WC FOR A PORT
030	001114	000000	AOB1: 0	:FLAG FOR PORT BEING TESTED
031	001116	000000	VADDR: 0	:VIRTUAL ADDR
032	001120	000000	PHADDR: 0	:PHYSICAL ADDR
033	001122	000000	FLAG2: 0	:FLAG FOR RESTART AND FOUND DRIVE
034	001124	000000	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

035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070

```

871                                     ;RH11 DEDICATE REGISTERS (MEMORY)
872
873 001126 000000 FLAG: 0 ;TEST REGISTER
874 001130 000000 WRDCT: 0 ;WORKING WORD COUNT
875 001132 000000 TRACK: 0 ;WORKING DAE
876 001134 000000 DMA: 0 ;WORKING DAR
877 001136 000000 PATNU: 0 ;DATA PATTERN INDEX
878 001140 000000 BUF: 0 ;WORKING DATA BUFFER (OUT-IN)
879 001142 000000 TDMA: 0 ;TEMP DAR
880 001144 000000 SWRDCT: 0 ;STANDARD WORD COUNT
881 001146 000000 ERCOUNT: 0 ;ERROR COUNT FOR MESSAGES.
882 001150 000000 SAVE: 0
883 001152 000000 HRDER: 0 ;POINTER FOR HARD ERROR
884 001154 000000 BLOCK: 0
885 001156 000000 PASSC: 0
886 001160 000000 UNNUM: 0 ;UNIT CURRENTLY BEING TESTED
887 001162 000000 UNITSV: 0 ;SET BIT=UNIT ON BUS
888 001164 000000 UNCMP: 0 ;FOR COMPARING FOR # OF DEVICE
889 001166 000000 RS04DT: 0 ;FLAG FOR RS04
890 001170 000000 NUMS: 0 ;WORK LOC FOR NUMBER INPUTS
891 001172 000000 CMD: 0 ;LOC FOR CS2 COMMANDS
892 001174 000000 SWITCH: 0 ;FLAG FOR WHICH RANDOM NUMBER GEN
893 001176 000000 INTFLG: 0 ;FLAG FOR INTERRUPT
894 001200 000000 LOPCNT: 0 ;ERROR FLAG AND LOOP COUNTER FLAG
895 001202 000000 WRITER: 0 ;CONTAINS # OF WRITE ERRORS
896 001204 000000 WCERR: 0 ;CONTAINS # OF WRITE CHECK ERRORS
897 001206 000000 READER: 0 ;CONTAINS # OF READ ERRORS
898 001210 000000 COMERR: 0 ;CONTAINS # OF COMPARE ERRORS
899 001212 000000 MMAVA: 0 ;MEM MGMT AVAILABLE INDICATOR
900 001214 000000 SAVWC: 0 ;SAVE LOC FOR CONVERSATION WC ROUTINE
901 001216 000000 FLAG3: 0 ;LOOP IN ADDRESS + RANDOM TST FLAG
902 001220 000000 SAVWCB: 0 ;SAVE WC SIZE FOR -B- PORT
903
904                                     ;RH11 WORK REGISTERS
905                                     ; (CAN BE CHANGED IN ANY ROUTINE)
906 001222 000000 WORK: 0
907 001224 000000 WORK1: 0
908 001226 000000 WORK2: 0
  
```

K02

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 24
 \$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0023

909	001230	005037	020116			SETSWI: CLR	SWI	
910	001234	012706	000500			BEGIN: MOV	#500, SP	:SET STACK TO *** 500 ***
911	001240	012737	016274	000024		MOV	#.POWER, @#24	:SET UP PF VECTOR
912	001246	012737	000340	000026		MOV	#340, @#26	:LOCK OUT THE WORLD
913	001254	012737	015736	000030		MOV	#.HLT, @#30	:SET EMT VECTOR
914	001262	012737	000340	000032		MOV	#340, @#32	:LOCK UP
915	001270	012737	016642	000034		MOV	#.TRAP, @#34	:SET TRAP VECTOR
916	001276	012737	000340	000036		MOV	#340, @#36	:LOCK UP
917	001304	005037	001000			CLR	ICNT	:INIT ICNT
918	001310	005037	001010			CLR	LAD	:INIT LAD
919	001314	023737	000042	000046		CMP	@#42, @#46	:UNDER ACT11 AUTO MODE?
920	001322	001423				BEQ	6\$:YES-SKIP TITLE PRINT-OUT
921	001324	104402	001330			TYPE	..+2	:.ASCIZ <15><12>"CZRSCG RH11-RS03/4 DA RELIAB"<15><12>
922	001372	032737	000001	020116	6\$:	BIT	#BIT0, SWI	
923	001400	001001				BNE	2\$	
924	001402	104444				SUSWR		:SIZE FOR SWITCHLESS
925	001404	042737	177677	001126	2\$:	BIC	#177677, FLAG	:CLEAR FLAG
926	001412	042737	177776	001122		BIC	#177776, FLAG2	:CLEAR ALL EXECPT RESTART
927	001420	005037	001216			CLR	FLAG3	:CLEAR LOOP IN ADDRESS + RANDOM TST FLAG
928	001424	032737	000001	001122		BIT	#BIT0, FLAG2	:IS THIS THE FIRST TIME?
929	001432	001002				BNE	1\$:NO
930	001434	004737	020120			JSR	PC, LDR	:SAVE LOADER
931	001440	000005			1\$:	RESET		:CLEAR THE WORLD
932	001442	012737	000340	177776		MOV	#340, PS	:LOCK UP INTERRUPT LEVELS
933	001450	004537	012452			JSR	RS, ERRCL	:CLEAR ERROR COUNTER + PASS CNT
934	001454	005037	001212			CLR	MMAVA	:CLEAR MEM MGMT FLAG
935	001460	005037	001114			CLR	A0B1	:TEST A PORT FIRST
936	001464	032777	000004	177334		BIT	#BIT2, @SWR	:WANT MEM MGMT?
937	001472	001021				BNE	3\$:NO
938	001474	012737	001522	000004		MOV	#5\$, 4	:SET TIMEOUT TRAP
939	001502	012737	000340	000006		MOV	#340, 6	:SET PS
940	001510	005037	177572			CLR	@#SR0	:IS MEM MGMT AVAILABLE?
941	001514	005137	001212			COM	MMAVA	:YES
942	001520	000401				BR	4\$:CONT
943	001522	022626			5\$:	CMP	(6)+, (6)+	:CLEAR STACK
944	001524	012737	000006	000004	4\$:	MOV	#6, 4	:RESET
945	001532	005037	000006			CLR	6	:TRAP
946	001536	032737	000001	001122	3\$:	BIT	#BIT0, FLAG2	:IS THIS THE FIRST TIME
947	001544	001002				BNE	CALM	:NO
948	001546	000137	020210			JMP	SIZZAP	:SIZE BUFFERS
949	001552	004737	011476			JSR	PC, @#EXTMEM	:SET UP DATA BUFFERS
950	001556	004737	015222			JSR	PC, MAMK	:TURN ON PARITY MEM
951	001562	032737	000001	001122		BIT	#BIT0, FLAG2	:1ST TIME ?
952	001570	001006				BNE	3\$:NO
953	001572	013737	001144	001214		MOV	SWRDC, SAVWC	:SAVE WC FOR CONVERSATION MODE COMPARE
954	001600	013737	001112	001220		MOV	WDCTB, SAVWCB	:SAVE WC FOR -B- PORT
955	001606	052737	000001	001122	3\$:	BIS	#BIT0, FLAG2	:SET 1ST TIME FLAG
956	001614	005037	001134			CLR	DMA	:CLEAR DAR REGISTERS
957	001620	005037	001136			CLR	PATNU	:CLEAR PATTEN COUNT
958	001624	013737	001144	001130		MOV	SWRDC, WRDCT	
959	001632	032777	000002	177166		BIT	#BIT1, @SWR	:DATA TEST ONLY?
960	001640	001403				BEQ	2\$:NO
961	001642	052737	002000	001126		BIS	#BIT10, FLAG	:YES
962	001650	032777	002000	177150	2\$:	BIT	#BIT10, @SWR	:ENTER CONVERSATION MODE?
963	001656	001007				BNE	1\$:YES GO TO CONVERSATION MODE
964	001660	052737	074000	001126		BIS	#74000, FLAG	

L02

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 25
\$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0024

965 001666 004537 010220
966 001672 000137 003232
967 001676 000137 002276

1\$: JSR R5,RESTOR ;RESTORE ORIGINAL WD CNT
JMP AD↑ST
JMP @#CONM ;ENTER CONVERSATION MODE

M02

```

968 ;FIND OUT HOW MANY DRIVES
969 ;FIRST TEST RSAS
970
971 001702 012701 000010          DRVENO: MOV      #8, R1          ;PUT 8 INTO R1 FOR COUNT
972 001706 042737 000002 001122    BIC      #BIT1, FLAG2        ;CLEAR FOUND DRIVE FLG
973 001714 012777 000000 177112    MOV      #0, @RSCS2         ;SET DEVICE TO ZERO
974 001722 012777 000007 177116    TRY:    MOV      #7, @RSER        ;CAUSE AN ERROR +SETS BIT IN AS REG
975 001730 005301 000000          DEC      R1                  ;DO A MAXIMUM OF 16 TIMES
976 001732 001403          BEQ      DVNUM              ;TESTED FOR ALL DRIVES GET OUT
977 001734 005277 177074          INC      @RSCS2            ;INCREMENT DRIVE UNIT
978 001740 000770          BR       TRY                ;REPEAT FOR NEXT DRIVE
979 001742 017737 177102 001162    DVNUM:  MOV      @RSAS, UNITSV ;SAVE
980 001750 043737 001124 001162    BIC      DROP, UNITSV       ;DROP BAD DRIVES
981 001756 012737 000401 001164    MOV      #401, UNCOMP       ;SETUP TO CMP WITH UNITSV
982 001764 012737 000000 001160    MOV      #0, UNNUM          ;PUT 0 INTO UNIT NO.
983 001772 032777 000040 177026    BIT      #BITS, @SWR        ;INHIBIT TYPE OUT?
984 002000 001005          BNE     STTEST             ;YES
985 002002 104402 000706          TYPE    , TSTNG            ;
986 002006 042737 000004 001122    BIC      #BIT2, FLAG2        ;CLEAR ERROR FLAG
987 002014 033737 001164 001162    STTEST: BIT      UNCOMP, UNITSV ;IS THIS DRIVE ON THE SYSTEM
988 002022 001473          BEQ     TRYNX              ;NO
989 002024 013777 001160 177002    UNTYP:  MOV      UNNUM, @RSCS2 ;YES PUT UNIT # INTO CS2
990 002032 005037 001166          CLR     RS04DT             ;CLEAR DRIVE TYPE FLAG
991 002036 022777 000004 177014    CMP     #4, @RS0DT         ;RS03LA?
992 002044 001004          BNE     8$                 ;NO
993 002046 012737 000004 001166    MOV     #4, RS04DT         ;SET DRIVE TYPE FLAG
994 002054 000422          BR      1$                 ;CONT
995 002056 005777 176776          8$:    TST     @RS0DT         ;IS THIS A RS03?
996 002062 001417          BEQ     1$                 ;YES
997 002064 022777 000001 176766    2$:    CMP     #1, @RS0DT     ;IS THIS A RS03 4US?
998 002072 001413          BEQ     1$                 ;YES
999 002074 022777 000002 176756    3$:    CMP     #2, @RS0DT     ;IS THIS A RS04?
1000 002102 001404          BEQ     6$                 ;YES
1001 002104 022777 000003 176746    CMP     #3, @RS0DT         ;RS04?
1002 002112 001037          BNE     TRYNX              ;GET A NEW NUMBER
1003 002114 052737 177777 001166    6$:    BIS     #-1, RS04DT     ;YES RS04
1004 002122 032737 040000 001122    1$:    BIT      #BIT14, FLAG2    ;IN POWER FAIL OR CONVERSATION?
1005 002130 001401          BEQ     7$                 ;NO
1006 002132 000207          RTS     PC                  ;YES
1007 002134 032777 000200 176702    7$:    BIT      #BIT7, @RS0DS    ;IS THIS DRIVE READY ?
1008 002142 001423          BEQ     TRYNX              ;NO GET ANOTHER DRIVE
1009 002144 032777 000040 176654    BIT     #BITS, @SWR        ;TYPEOUT?
1010 002152 001016          BNE     4$                 ;NO
1011 002154 032737 000004 001122    BIT     #BIT2, FLAG2       ;WAS THERE AN ERRER?
1012 002162 001402          BEQ     5$                 ;NO
1013 002164 104402 000636          TYPE    , CRLFLF          ;
1014 002170          5$:
1015 002170 013746 001160          MOV     UNNUM, -(6)        ;PUT UNNUM ON STACK
1016 002174 104406          TYFES  , 40                ;TYPE STACK IN OCTAL - SUPRESS
1017 002176 104402 000040          TYPE    , 40                ;TYPE SPACE
1018 002202 042737 000004 001122    BIC     #BIT2, FLAG2       ;CLEAR ERROR FLAG
1019 002210 000426          BR      NOWGO              ;NOW TEST
    
```

N02

CZRSCG RH11-R503-R503/LA-R504 DATA AND RELIABILITY TEST
 CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 27
 \$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0026

```

1020 002212 006337 001164      TRYNX:  ASL      UNCMP      ;CHECK NEXT BIT FOR DRIVE
1021 002216 103403              BCS      CHCKDV     ;DID WE TEST ANY REG?
1022 002220 005237 001160      INC      UNNUM      ;INC UNIT #
1023 002224 000673              BR       STTEST     ;CHECK FOR NEXT DRIVE
1024
1025      ;THIS PROGRAM WILL DEFAULT TO TESTING UNIT 0 IF IT CAN NOT FIND ANY DRIVES
1026 002226 032737 000002 001122  CHCKDV:  BIT      #BIT1,FLAG2 ;FOUND DRIVE?
1027 002234 001012              BNE      DONEE     ;YES WE DID TEST A DRIVE
1028 002236 012737 100000 001164  MOV      #100000,UNCMP ;NO DRIVES TESTED, COULD NOT SET
1029 002244 005037 001160      CLR      UNNUM      ;ANY AS BITS, THUS DEFAULTS TO 0
1030 002250 013746 001160      MOV      UNNUM,-(6) ;PUT UNNUM ON STACK
1031 002254 104406              TYPES     ;TYPE STACK IN OCTAL - SUPPRESS
1032 002256 000000              HALT      ;COULD NOT SET ANY ATA BITS
1033
1034      ;BY SETTING ERROR BITS
1035 002260 000402              BR       NOWGO     ;GO BACK AND USE OTHER DIAG.
1036 002262 000137 013466 001122  DONEE:  JMP      OUT       ;DEFAULT TO DRIVE 0
1037 002266 052737 000002 001122  NOWGO:  BIS      #BIT1,FLAG2 ;GET OUT
1038 002274 000207              RTS      PC        ;FOUND DRIVE
1039
1040      ;ENTER OPERATOR CONVERSATION MODE
1041
1042 002276              CONM:
1043 002276 104402 002302              TYPE     .+2      ;.ASCIZ <15><12>"-A- PORT"
1044 002316 004737 003306              JSR      PC,CMPLY ;COMPARE FOR YES
1045 002322 001405              BEQ      2$       ;YES
1046 002324 012737 177777 001114  MOV      #-1,AOB1 ;B PORT
1047 002332 000137 002354              JMP      1$       ;TEST -B- PORT
1048 002336 104402 000653              2$:     TYPE     ,STABUF
1049 002342 104420              RDOCT
1050 002344 012637 001070      MOV      (6)+,STAMEM ;START BUFFER AT 4K
1051 002350 000137 002374              JMP      NOPORT   ;GET OUT
  
```

B03

CZRSCG RH11-RS03-RS03,LA-RS04 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 28
SKMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEG 0027

1052	002354	104402	000653		1S:	TYPE	.STABUF	
1053	002360	104420				RDOCT		:GET ANS
1054	002362	012637	001074			MOV	(6)+,STBCOM	:AND SAVE IT
1055	002366	052737	000100	001126		BIS	#BIT6,FLAG	:SET B PORT FLAG
1056	002374	004737	011476		NOPORT:	JSR	PC,EXTMEM	:CAL BUFFERS AND WC
1057	002400	013737	001144	001130		MOV	SWROCT,WRDCT	:GET STANDARD WC
1058	002406	052737	002000	001126	A1:	BIS	#BIT10,FLAG	:SET BIT FOR DATA TEST ONLY
1059	002414	004537	012452			JSR	RS,ERRCL	:CLEAR ERROR CNT + PASS CNT
1060	002420	042737	174040	001126		BIC	#174040,FLAG	:CLEAR MULTI FLAG MODE +PATTERN SELECT
1061	002426	104402	002432			TYPE	.+2	:ASCIZ <15><12>"MULTI DRIVE"
1062	002450	004737	003306			JSR	PC,CMPLY	:COMPARE FOR YES
1063	002454	001004				BNE	DATTES	:ANS IS NO
1064	002456	052737	004000	001126		BIS	#BIT11,FLAG	:SET BIT FOR MULTI DRIVE
1065	002464	000434			1S:	BR	ASKWC	
1066	002466	104402	000510		DATTES:	TYPE	.LOADSW	
1067	002472	104420				RDOCT		
1068	002474	012637	001170			MOV	(6)+,NUMS	:GET NUMBER
1069	002500	022737	000010	001170		CMP	#10,NUMS	:CORRECT # ?
1070	002506	103767				BLO	DATTES	:NO
1071	002510	013737	001170	001160		MOV	NUMS,UNNUM	:SET UNIT #
1072	002516	004737	006544			JSR	PC,FNDTYP	:TEST FOR RS04 OR 03
1073	002522	005002			1S:	CLR	R2	:CLEAR WORK AREA
1074	002524	000261				SEC		:SET CARRY
1075	002526	006102			2S:	ROL	R2	:SET BIT IN WORK
1076	002530	005737	001170			TST	NUMS	:IS THIS THE RIGHT BIT FOR THE RIGHT DISK
1077	002534	001403				BEQ	3S	:YES
1078	002536	005337	001170			DEC	NUMS	:NO TRY AGAIN
1079	002542	000771				BR	2S	:TEST AGAIN
1080	002544	010237	001162		3S:	MOV	R2,UNITSV	:SET DRIVE BIT IN UNITSV
1081	002550	052737	000002	001122		BIS	#BIT1,FLAG2	:SET FOUND DRIVE FLAG
1082								
1083	002556				ASKWC:	TYPE	.+2	:ASCIZ <15><12>"OPT WD CT"
1084	002556	104402	002562			JSR	PC,CMPLY	:COMPARE FOR YES
1085	002576	004737	003306			BEG	WCCON	:YES
1086	002602	001401				BR	OPDAR	:CONT
1087	002604	000444						

1088	002606					WCCON:	TYPE	..+2	;.ASCIZ <15><12>"WD CT "
1089	002606	104402	002612				RDOCT		
1090	002624	104420					MOV	(6)+,NUMS	;GET NUMBER
1091	002626	012637	001170				TST	NUMS	;IS IT 0?
1092	002632	005737	001170				BEG	WCCON	;YES ASK AGAIN FOR LENGTH
1093	002636	001763					MOV	SAVWC,R2	;GET STANDARD WC FOR -A- PORT
1094	002640	013702	001214				TST	AOB1	;-A- PORT?
1095	002644	005737	001114				BEG	1\$;YES
1096	002650	001402					MOV	SAVWCB,R2	;GET WC FOR -B- PORT
1097	002652	013702	001220			1\$:	INC	R2	
1098	002656	005202					CMP	R2,NUMS	;IS NUMS LESS THAN SWRDCT
1099	002660	020237	001170				BLOS	WCCON	;YES ASK FOR COUNT AGAIN
1100	002664	101750					MOV	NUMS,SWRDCT	;OPERATING WORD COUNT
1101	002666	013737	001170	001144			MOV	SWRDCT,WRDCT	
1102	002674	013737	001144	001130			TST	AOB1	;B PORT?
1103	002702	005737	001114				BEG	OPDAR	;NO
1104	002706	001403					MOV	WRDCT,WDCTB	;YES GET WC
1105	002710	013737	001130	001112					
1106							OPDAR:	TYPE	;.OPDR
1107	002716	104402	000535				JSR	PC,COMPY	;COMPARE FOR YES
1108	002722	004737	003306				BNE	OPPAT	;ANS IS NO
1109	002726	001037					BIS	#BITS,FLAG	;SET OPTIONAL DMA FLAG
1110	002730	052737	000040	001126			TYPE	..+2	;.ASCIZ ""
1111	002736	104402	002742				RDOCT		
1112	002744	104420					MOV	(6)+,NUMS	;GET NUMBER
1113	002746	012637	001170				CMP	#4,R504DT	;R503LA?
1114	002752	022737	000004	001166			BNE	3\$;NO
1115	002760	001004					CMP	#3777,NUMS	;IS ADDR. CORRECT?
1116	002762	022737	003777	001170			BR	2\$	
1117	002770	000412					TST	R504DT	;IS THIS A R504?
1118	002772	005737	001166			3\$:	BEG	1\$;NO
1119	002776	001404					CMP	#17777,NUMS	;IS ADD. CORRECT?
1120	003000	022737	017777	001170			BR	2\$;GET OUT
1121	003006	000403					CMP	#7777,NUMS	;IS ADD. CORRECT?
1122	003010	022737	007777	001170		1\$:	BLOS	OPDAR	;NO
1123	003016	101737				2\$:	MOV	NUMS,TDMA	;TEMP SECTOR REGISTER
1124	003020	013737	001170	001142					

1125	003026				OPPAT:	TYPE	,.+2	;.ASCIZ <15><12>"PATN # "
1126	003026	104402	003032			RDOCT		
1127	003044	104420				MOV	(6)+ NUMS	;.GET NUMBER
1128	003046	012637	001170			CMP	#23, NUMS	;.TEST FOR CORRECT NO
1129	003052	022737	000023	001170		BLOS	OPPAT	;.ASK AGAIN
1130	003060	101762				CLR	PATNU	;.CLEAR PATTERN #
1131	003062	005037	001136			CMP	#22, NUMS	
1132	003066	022737	000022	001170		BEQ	OPWRT	;.DATA PATTERN UNDER PROGRAM CONTROL
1133	003074	001411				BIS	#BIT15, FLAG	;.SET PROGRAM FLAG
1134	003076	052737	100000	001126		MOV	NUMS, PATNU	;.OPERATOR WANTS TO SELECT DATA
1135	003104	013737	001170	001136		CLC		
1136	003112	000241				ROL	PATNU	
1137	003114	006137	001136					
1138								
1139	003120				OPWRT:	TYPE	,.+2	;.ASCIZ <15><12>"WRITE"
1140	003120	104402	003124			JSR	PC, CMPY	;.COMPARE FOR YES
1141	003134	004737	003306			BNE	OPRD	;.ASK ABOUT WRITE CHECK
1142	003140	001003				BIS	#BIT14, FLAG	;.YES SET FLAG BIT
1143	003142	052737	040000	001126				
1144								
1145	003150				OPRD:	TYPE	,.+2	;.ASCIZ <15><12>"READ"
1146	003150	104402	003154			JSR	PC, CMPY	;.COMPARE FOR YES
1147	003164	004737	003306			BNE	OPWCK	
1148	003170	001003				BIS	#BIT12, FLAG	;.SET FLAG TO READ
1149	003172	052737	010000	001126				
1150								
1151	003200				OPWCK:	TYPE	,.+2	;.ASCIZ <15><12>"WRT CK"
1152	003200	104402	003204			JSR	PC, CMPY	;.COMPARE FOR YES
1153	003216	004737	003306			BNE	ADTST	
1154	003222	001003				BIS	#BIT13, FLAG	
1155	003224	052737	020000	001126				
1156								
1157	003232	032737	004000	001126	ADTST:	BIT	#BIT11, FLAG	;.ARE WE IN MULTI DRIVE MODE
1158	003240	001402				BEQ	EXMFLG	;.BRANCH IF NO.
1159	003242	004737	001702			JSR	PC, DRVENO	;.GET DRIVES TO BE TESTED
1160	003246	042737	000004	001126	EXMFLG:	BIC	#BIT2, FLAG	;.CLEAR XFER MODE FLAG
1161	003254	032777	000400	175544		BIT	#BIT8, JSWR	;.XFER MODE?
1162	003262	001403				BEQ	1\$;.NO
1163	003264	052737	000004	001126		BIS	#BIT2, FLAG	;.SET XFER MODE FLAG
1164	003272	032737	002000	001126	1\$:	BIT	#BIT10, FLAG	;.TEST FOR DATA TEST ONLY
1165	003300	001414				BEQ	ADT1	;.DO COMPLETE TEST
1166	003302	000137	003716			JMP	DATAT	;.DO DATA TEST ONLY
1167								
1168	003306	104402	000641		CMPY:	TYPE	,YORN	
1169	003312	104422				RDLIN		
1170	003314	122737	000131	017052		CMPB	#'Y, INPUT	;.TEST FOR YES
1171	003322	000207				RTS	PC	
1172	003324	052737	100000	001216	ADTL:	BIS	#BIT15, FLAG3	;.SET LOOP IN ADDRESS TEST FLAG GOT HERE
1173								;.BECAUSE PROGRAM WAS STARTED AT 234

E03

CZRSCG RH11-R503-R503-LA-R504 DATA AND RELIABILITY TEST
 CZRSCG.F11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 31
 SKMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0030

```

1174 :RH11 ADDRESS TEST #1 (TRACK AND SECTOR SELECTION TEST)
1175 :WRITE 100(OCTAL) R503, 200(OCTAL) R504, WORDS IN EACH SECTOR
1176 :THE WORD CONTAINS THE ADDRESS OF EACH SECTOR
1177 :WHEN THE COMPLETE DISK IS WRITTEN READ
1178 :BACK EACH SECTOR AND COMPARE FOR THE CORRECT
1179 :DATA IN THE SECTOR
1180 :PS IS AT LEVEL 7 SO NO INTERRUPTS
1181
1182 003332 ADT1: ;ADDRESS TEST
1183 :*****
1184 :TEST 1 ADDRESS TEST
1185 :*****
1186 003332 104400 TST1: SCOPE
1187 003334 032737 000004 001126 ADT1A: BIT #BIT2,FLAG ;XFER MODE?
1188 003342 001402 3$ BEQ 3$ ;NO
1189 003344 000137 003716 JMP DATAT ;YES
1190 003350 012737 000340 177776 3$: MOV #340,PS ;LOCK UP PS
1191 003356 012737 020000 017436 MOV #20000,OUTBUF ;START BUF AT 20000
1192 003364 052737 000400 001126 BIS #BIT8,FLAG ;SET TEST FLAG
1193 003372 013737 001144 001150 MOV SWRDCT,SAVE ;SAVE STD WD COUNT
1194 003400 005037 001134 CLR DMA ;CLEAR DISK ADD
1195 003404 104426 CLRDV ;INIT DRIVE
1196 003406 004737 006566 JSR PC,WHTHU ;GET WORD COUNT
1197 003412 013737 001130 001144 5$: MOV WRDCT,SWRDCT
1198 003420 013737 017436 001140 2$: MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
1199 003426 104414 SEABUF: ERCLR ;CLEAR RS REGISTERS IF ERROR
1200 003430 013700 017436 MOV OUTBUF,R0 ;SET UP ADDRESS BUFFER
1201 003434 013701 001130 MOV WRDCT,R1
1202 003440 013720 001134 XSEABUF: MOV DMA,(0)+ ;LOAD OUTBUF WITH DATA TO BE WRITTEN
1203 003444 005301 DEC R1 ;FILL OUTBUF
1204 003446 001374 BNE XSEABUF ;WITH DATA
1205 003450 012737 000061 001172 MOV #61,CMD ;WRITE NO I/E
1206 003456 104416 DKCMD ;GO WRITE
1207 003460 105777 175346 TSTB @RSCS1 ;CHECK FOR READY
1208 003464 100375 BPL -4
1209 003466 005777 175340 TST @RSCS1 ;TEST FOR ERROR
1210 003472 100010 BPL WRNEXB ;BRANCH IF NO ERROR
1211 003474 012737 003426 001010 MOV #SEABUF,LAD ;SET UP LOOP ADDRESS
1212 003502 052737 001000 001126 BIS #BIT9,FLAG ;SET ERROR BIT IN FLAG
1213 003510 104430 LOGW ;LOG WRITE ERROR
1214 003512 104034 HLT !WC!DA!BA
1215 003514 104400 WRNEXB: SCOPE
1216 003516 004737 007054 JSR PC,DISBUF ;SET UP NEXT DISK ADDR.
1217 003522 000741 BR SEABUF ;WRITE NEXT SECTOR
1218 003524 104400 RRDSEC: SCOPE
  
```

```

1219 003526 104414 RDSECT: ERCLR          :CLEAR ERRORS
1220 003530 012737 000071 001172 MOV          #71,CMD      :READ NO I/E
1221 003536 104416 DKCMD        :DO A READ
1222 003540 105777 175266 TSTB         @RSCS1     :CHECK FOR READY
1223 003544 100375 BPL          -4        :NOT READY BRANCH BACK
1224 003546 005777 175260 TST          @RSCS1     :TEST FOR ERROR
1225 003552 100006 BPL          ADHGT      :BRANCH IF NO ERROR
1226 003554 052737 001000 001126 BIS          #BIT9,FLAG :SET ERROR FLAG
1227 003562 104432 LOGR          :LOG READ ERROR
1228 003564 104014 HLT          !WC!DA
1229 003566 104400 SCOPE
1230 003570 013702 017436 ADHGT: MOV      OUTBUF,R2
1231 003574 013746 001130 MOV      WRDCT, -(SP)   :SAVE OLD WC
1232 003600 004737 006566 JSR      PC,WHTHU
1233 003604 013703 001130 MOV      WRDCT,R3
1234 003610 012637 001130 MOV      (SP)+,WRDCT  :RESTORE OLD WC
1235 003614 023712 001134 SANHT: CMP      DMA,(2)  :CMP FOR CORRECT ADDR.
1236 003620 001004 BNE      ADERR        :BRANCH IF DATA DID NOT COMPARE
1237 003622 005722 TST      (2)+         :GET NEXT ADDRESS OF INBUF
1238 003624 005303 DEC      R3           :DEC SECTOR COUNT
1239 003626 001372 BNE      SANHT        :TEST NEXT WORD
1240 003630 000412 BR
1241 003632 013701 001134 ADERR: MOV      DMA,GOOD :CORRECT ADDRESS
1242 003636 011200 MOV      (2),BAD      :DATA IN ERROR
1243 003640 104000 HLT          :DISK ADD DID NOT MATCH WRITTEN ADDRESS
1244 003642 104436 LOGC
1245 003644 004737 014224 JSR      PC,PRNT      :LOG COMPARE ERROR
1246 003650 001002 BNE      CHKADT      :INHIBIT TYPEOUT?
1247 003652 104402 000636 TYPE      ,CRLF      :YES
1248
1249
1250 ;*****REPORT ONLY ONE ERROR PER SECTOR*****
1251 003656 104400 CHKADT: SCOPE
1252 003660 004737 007054 JSR      PC,DISBUF   :SET UP NEXT DISK BUFFER
1253 003664 000717 BR          RRDSEC    :CHECK NEXT SECTOR
1254 003666 013737 001150 001144 MOV      SAVE,SWRDCT :GET STD WD COUNT
1255 003674 042737 000400 001126 BIC      #BIT8,FLAG  :CLEAR TEST FLAG
1256 003702 032737 100000 001216 BIT      #BIT15,FLAG3 :DOES OPERATOR WANT TO LOOP ON TEST
1257 003710 001402 BEQ      .+6         :NO
1258 003712 000137 003334 JMP      ADT1A       :YES
    
```



```

1259 003716          DATAT:          ;DATA TEST
1260          ;*****
1261          ;TEST 2          DATA TEST
1262          ;*****
1263 003716 104400          †TST2:  SCOPE
1264 003720 005037 001134          CLR          DMA          ;CLEAR DISK ADDRESS
1265 003724 104426          CLRDV          ;CLEAR RS REGISTERS
1266 003726 013737 001144 001130          MOV          SWRDCT,WRDCT
1267 003734 032737 000100 001126          BIT          #BIT6,FLAG          ;MULTI PORT?
1268 003742 001403          BEQ          3$          ;NO
1269 003744 005737 001114          TST          AOB1          ;A OR B PORT?
1270 003750 001003          BNE          1$          ;B PORT
1271 003752 004737 011354          3$:  JSR          PC,APORT          ;A PORT
1272 003756 000402          BR          2$
1273 003760 004737 011422          1$:  JSR          PC,BPORT          ;B PORT
1274 003764 004737 007000          2$:  JSR          PC,VECTRR          ;SETUP INT VECTOR
1275 003770 012777 000340 175066          MOV          #340,STATUS          ;SET DISK STATUS REG LOC (206)
1276 003776 012737 004030 001152          MOV          #LDAT,HRDR          ;SETUP FOR HARD ERROR RETURN
1277 004004 013737 017436 001140          MOV          OUTBUF,BUF          ;SETUP OUTPUT BUFFER
1278 004012 052737 000003 001126          BIS          #3,FLAG          ;SET COUNTER TO 3
1279 004020 004537 007722          JSR          RS,PASEL          ;SET UP DATA BUFFERS
1280 004024 005037 001200          LDAT1: CLR          LOPCNT          ;CLEAR ERROR FLAG
1281 004030 104414          LDAT:  ERCLR          ;CLEAR RS REG. IF ERROR
1282 004032 004537 006504          JSR          RS,OPDSEL          ;SET UP DISK ADDRESS
1283 004036 032737 040000 001126          BIT          #BIT14,FLAG          ;TEST FOR WRITE
1284 004044 001456          BEQ          SLH          ;TEST FOR READ
1285 004046 012737 000161 001172          MOV          #161,CMD          ;WRITE WITH I/E
1286 004054 104416          DKCMD          ;DO A WRITE
1287 004056 004737 011770          JSR          PC,WATT          ;WAIT FOR INTERRUPT
1288 004062 012737 004030 001010          MOV          #LDAT,LAD          ;SETUP SCOPE LOOP
1289 004070 032737 001000 001126          BIT          #BIT9,FLAG          ;WAS THERE AN ERROR?
1290 004076 001423          BEQ          WRXBL          ;CONT
1291 004100 104430          LOGW          ;LOG WRITE ERROR
1292 004102 005237 001200          INC          LOPCNT          ;SET ERROR FLAG
1293 004106 004737 014224          2$:  JSR          PC,PRNT          ;TYPE ?
1294 004112 001004          BNE          1$          ;NO
1295 004114 104402 000554          TYPE          ,DATA
1296 004120 104402 000564          TYPE          ,WRTErr
1297 004124 104044          1$:  HLT          !DS!DA          ;WRITE ERROR
1298 004126 005337 001126          DEC          FLAG          ;DEC COUNTER
1299 004132 032737 000003 001126          BIT          #3,FLAG          ;DONE YET WITH 3RD TRY?
1300 004140 001333          BNE          LDAT          ;NOT 3 TRIES YET? TRY AGAIN
1301 004142 004737 011460          JSR          PC,WTNO          ;TYPE CAN NOT WRITE

```


1345	004376	004737	006632			JSR	PC,PHYCOV	:YES GET VITURAL ADDR
1346	004402	010237	001222		7\$:	MOV	R2,WORK	:GET BA
1347	004406				8\$:			
1348	004406	104402	004412			TYPE	..+2	:.ASCIZ <15><12>"(BA)=""
1349	004422				6\$:			
1350	004422	017746	174574			MOV	@WORK,-(6)	:PUT @WORK ON STACK
1351	004426	104404				TYPEO		:TYPE STACK IN OCTAL
1352	004430	104402	004434			TYPE	..+2	:.ASCIZ "WC=""
1353	004442	017746	174370			MOV	@RSWC,-(6)	:PUT @RSWC ON STACK
1354	004446	104404				TYPEO		:TYPE STACK IN OCTAL
1355	004450	104026			4\$:	HLT	!DA!DB!BA	:NOTE: BA REG. = +2 OF ACTUAL MEMORY
1356								:LOC AFTER WORDS HAVE BEEN XFERED
1357	004452	005237	001200		10\$:	INC	LOPCNT	:INC ERROR COUNT
1358	004456	022737	000010	001200		CMP	#10,LOPCNT	:ID TRYS YET?
1359	004464	001247				BNE	SLH2	:NO
1360	004466	004737	006524			JSR	PC,NOREC	:TYPE UNRECOVERABLE
1361	004472	005737	001200		1\$:	TST	LOPCNT	:ANY ERRORS?
1362	004476	001402				BEQ	5\$:NO
1363	004500	004737	012012			JSR	PC,TYPREC	:TYPE RECOVERED
1364	004504	005037	001200		5\$:	CLR	LOPCNT	:CLEAR ERROR COUNTER
1365	004510	104400				SCOPE		
1366	004512	012737	004204	001010		MOV	#SLH2,LAD	:SETUP LOOP ADDRESS
1367	004520	004737	007054			JSR	PC,DISBUF	:SET UP THE DISK BUFFER
1368	004524	000422				BR	SLH2A	
1369	004526	004537	011306		ESH1:	JSR	R5,CLEAR	:CLEAR BUFFER
1370	004532	004537	006504		ESH:	JSR	R5,OPDSEL	:OPERATOR SELECTED DISK ADDRESS?
1371	004536	032737	010000	001126		BIT	#BIT12,FLAG	:TEST FOR READ
1372	004544	001002				BNE	1\$:YES
1373	004546	000137	004760			JMP	MSTR	:NO READ
1374	004552	104400			1\$:	SCOPE		
1375	004554	042737	000003	001126		BIC	#3,FLAG	:CLEAR RE-READ COUNT
1376	004562	005037	001200			CLR	LOPCNT	:CLEAR FLAG
1377	004566	000137	004576			JMP	DSKRD	:CONT
1378	004572	000137	004204		SLH2A:	JMP	SLH2	
1379	004576	104414			DSKRD:	ERCLR		:CLEAR RS REG IF ERRORS
1380	004600	012737	000171	001172		MOV	#171,CMD	:READ WITH I/E
1381	004606	104416				DKCMD		:READ
1382	004610	004737	011770			JSR	PC,WATT	:WAIT FOR INTERRUPT
1383	004614	032777	010000	174204		BIT	#10000,@SWR	:COMPARE?
1384	004622	001006				BNE	ELH	:NO
1385	004624	032737	000004	001126		BIT	#BIT2,FLAG	:COMPARE?
1386	004632	001002				BNE	ELH	:NO
1387	004634	004537	010452			JSR	RS,COMPARE	:COMPARE

1388	004640	032737	001000	001126	ELH:	BIT	#BIT9,FLAG	: IS THERE AN ERROR?
1389	004646	001433				BEQ	ADRD	: NO
1390	004650	032777	000100	174150		BIT	#BIT6,ASWR	: SOFT ERROR TYPEOUT?
1391	004656	001003				BNE	3\$: YES
1392	004660	005737	001200			TST	LOPCNT	: FIRST ERROR?
1393	004664	001011				BNE	2\$: NO
1394	004666	004737	014224		3\$:	JSR	PC,FRNT	: TYPEOUT?
1395	004672	001004				BNE	1\$: NO
1396	004674	104402	000554			TYPE	,DATA	
1397	004700	104402	000607			TYPE	,RDERR	
1398	004704	104432			1\$:	LOGR		: LOG READ ERROR
1399	004706	104044				HLT	!DS!DA	
1400	004710	104400			2\$:	SCOPE		
1401	004712	005237	001200			INC	LOPCNT	: COUNT ERRORS
1402	004716	022737	000010	001200		CMP	#10,LOPCNT	: LAST RETRY?
1403	004724	001324				BNE	DSKRD	: NO
1404	004726	004737	006524			JSR	PC,NOREC	: TYPE UNRECOVERABLE
1405	004732	000137	004752		4\$:	JMP	ADR1	: CONT
1406	004736	104400			ADRD:	SCOPE		
1407	004740	005737	001200			TST	LOPCNT	: ANY ERRORS?
1408	004744	001402				BEQ	ADR1	: NO
1409	004746	004737	012012			JSR	PC,TYPREC	: TYPE RECOVERED
1410	004752	004737	007054		ADR1:	JSR	PC,DISBUF	: GO SET UP DISK BUFFER.
1411	004756	000434				BR	READCT	: CONT. READING
1412	004760	005737	001126		MSTR:	TST	FLAG	
1413	004764	100423				BMI	3\$: OPERATOR SELECTED PATTERN
1414	004766	062737	000002	001136		ADD	#2,PATNU	: INC PATTERN INDEX
1415	004774	022737	000044	001136		CMP	#44,PATNU	
1416	005002	001402				BEQ	.+6	
1417	005004	000137	003716			JMP	DATAT	: NOT LAST PATTERN EXIT
1418	005010	005037	001136			CLR	PATNU	: LAST PATTERN EXIT
1419	005014	032777	000002	174004		BIT	#BIT1,ASWR	: DATA TEST ONLY?
1420	005022	001006				BNE	2\$: YES
1421	005024	032737	002000	001126		BIT	#BIT10,FLAG	: DATA TEST ONLY?
1422	005032	001404				BEQ	1\$: NO
1423	005034	005137	001114		3\$:	COM	AOB1	: ALTERNATE PORTS
1424	005040	000137	006072		2\$:	JMP	EXTPPR	: LOOP
1425	005044	000137	005062		1\$:	JMP	RANEX	: DO NEXT TEST
1426								
1427	005050	000137	004532		READCT:	JMP	ESH	: CONT. READING
1428	005054	052737	100000	001216	RANEL:	BIS	#BIT15,FLAG3	: SET LOOP IN RANDOM TEST GOT
1429								: HERE BY STARTING AT LOC 240

K03

RANEX: ;RANDOM ADDRESS DATA TEST
;THIS PROGRAM WRITES, WRITECHECKS AND READS 1 SECTOR OF RANDOM DATA FROM RANDOM DISK
;ADDRESSES. THIS TEST WILL MAKE 1000(10) PASSES BEFORE IT IS COMPLETED

:TEST 3 RANDOM ADDRESS RANDOM DATA TEST

1430 005062
1431
1432
1433
1434
1435
1436
1437 005062 104400
1438 005064 032737 000004 001126
1439 005072 001402
1440 005074 000137 006072
1441 005100 052737 000400 001126
1442 005106 012737 020000 017436
1443 005114 013737 017436 001116
1444 005122 005737 001212
1445 005126 001402
1446 005130 005037 177572
1447 005134 012737 000042 001136
1448 005142 104426
1449 005144 012737 176030 001156
1450 005152 012737 005712 001152
1451 005160 004737 007000
1452 005164 012777 000340 173672
1453 005172 012737 005274 001010
1454 005200 012737 000001 001222
1455 005206 013701 017436
1456 005212 004537 010044
1457 005216 017737 012214 001134
1458 005224 042737 170000 001134
1459 005232 052737 000003 001126
1460 005240 004737 006566
1461 005244 013737 001130 001222
1462 005252 013701 017436
1463 005256 004537 010044
1464 005262 013737 017436 001140
1465 005270 005037 001200

TST3: SCOPE
BIT #BIT2, FLAG ;FAST XFER MODE?
BEQ 2\$;NO
JMP EXTPPR ;GET OUT
BIS #BIT8, FLAG ;SET TEST FLAG
MOV #20000, OUTBUF ;GET STARTING ADDR OF BUF
MOV OUTBUF, VADDR ;SAVE BUFFER ADDR
TST MMAVA ;MEM MGMT AVAILABLE?
BEQ 1\$;NO
CLR @#SRO ;TURN IT OFF
MOV #42, PATNU ;DO RANDOM COMPARE
CLRDV ;INIT DRIVE
MOV #-1000, PASSC ;SET UP PASS COUNT
MOV #RWRED, ARDR ;SET UP FOR HARD ERROR
JSR PC, VECTR ;SETUP INTERRUPT VECTOR
MOV #340, JSTATUS
WRLG1: MOV #WREAR, LAD ;SETUP LOOP ADDRESS
MOV #1, WORK ;SET UP RANDOM GENERATOR WORD
MOV OUTBUF, R1
JSR R5, RANDOM ;GENERATE RANDOM DATA
MOV @OUTBUF, DMA ;SET UP DISK ADDRESS
BIC #170000, DMA
BIS #3, FLAG ;SET COUNTER
JSR PC, WTHU ;GET WORD COUNT
MOV WRDCT, WORK ;GENERATE RANDOM BUFFER
MOV OUTBUF, R1
JSR R5, RANDOM
MOV OUTBUF, BUF ;SET UP OUTPUT BUFFER
CLR LOPCNT ;CLR ERROR FLAG

L03

1466	005274	104414			WRERR:	ERCLR			
1467	005276	012737	000161	001172		MOV	#161,CMD	:WRITE WITH I/E	
1468	005304	104416				DKCMD		:WRITE	
1469	005306	004737	011770			JSR	PC,WATT	:WAIT FOR INTERRUPT	
1470	005312	032737	001000	001126	2\$:	BIT	#BIT9,FLAG	:WAS THERE AN ERROR?	
1471	005320	001435				BEQ	WRRCK1	:NO	
1472	005322	032777	000100	173476		BIT	#BIT6,JSWR	:TYPE RETRY ?	
1473	005330	001003				BNE	5\$:YES	
1474	005332	005737	001200			TST	LOPCNT	:FIRST TIME?	
1475	005336	001013				BNE	6\$:NO	
1476	005340	104430			5\$:	LOGW		:LOG WRITE ERROR	
1477	005342	005237	001200			INC	LOPCNT	:SET ERROR FLAG	
1478	005346	004737	014224			JSR	PC,PRNT	:TYPEOUT?	
1479	005352	001004				BNE	3\$:YES	
1480	005354	104402	000674			TYPE	,RANDM		
1481	005360	104402	000564			TYPE	,WRERR		
1482	005364	104044			3\$:	HLT	!DS!DA		
1483	005366	104400			6\$:	SCOPE			
1484	005370	005337	001126			DEC	FLAG		
1485	005374	032737	000003	001126		BIT	#3,FLAG		
1486	005402	001334				BNE	WRERR	:RETRY	
1487	005404	004737	011460			JSR	PC,WTNO	:TYPE CAN NOT WRITE	
1488	005410	000137	006014			JMP	EXRAX	:GET NEW NUMBER	

M03

1489	005414	005737	001200		WRRCK1:	TST	LOPCNT		; ANY ERRORS?
1490	005420	001402				BEG	1\$; NO
1491	005422	004737	012012			JSR	PC, TYPREC		; TYPE RECOVERED
1492	005426	104400			1\$:	SCOPE			
1493	005430	005037	001200			CLR	LOPCNT		; CLEAR LOOP COUNT
1494	005434	104414			WRRCK:	ERCLR			; CLEAR RS REG IF ERRORS
1495	005436	012737	000151	001172		MOV	#151, CMD		; WRITE CHECK WITH I/E
1496	005444	104416				DKCMD			; WRITE CHECK
1497	005446	004737	011770			JSR	PC, WATT		; WAIT FOR INTERRUPT
1498	005452	032737	001000	001126	4\$:	BIT	#BIT9, FLAG		; ERROR?
1499	005460	001453				BEG	1\$; NO
1500	005462	032777	000100	173336		BIT	#BIT6, ASWR		; TYPE ALL RETRYS?
1501	005470	001003				BNE	2\$; YES
1502	005472	005737	001200			TST	LOPCNT		; FIRST ERROR?
1503	005476	001030				BNE	5\$; NO
1504	005500	104434			2\$:	LOGWC			; LOG WRITE CK
1505	005502	004737	014224			JSR	PC, PRNT		; TYPEOUT?
1506	005506	001052				BNE	6\$; NO
1507	005510	104402	000674			TYPE	, RANDM		
1508	005514	104402	000574			TYPE	, WCKERR		
1509	005520	017737	173314	001222		MOV	ASBBA, WORK		; GET CORRECT BA
1510	005526	162737	000002	001222		SUB	#2, WORK		
1511	005534	104402	005540			TYPE	, +2		; .ASCIZ <15><12>"(BA)=""
1512	005550	017746	173446			MOV	WORK, -(6)		; PUT WORK ON STACK
1513	005554	104404				TYPEO			; TYPE STACK IN OCTAL
1514	005556	104026				HLT	!DB!DA!BA		; BA=MEMORY LOC +2 OF ACTUAL WORD
1515	005560	005237	001200		5\$:	INC	LOPCNT		; INC RETRY COUNT
1516	005564	022737	000010	001200		CMP	#10, LOPCNT		; LAST ONE YET?
1517	005572	001320				BNE	WRRCK		; NO
1518	005574	104402	000522			TYPE	, UNRECO		
1519	005600	005037	001200			CLR	LOPCNT		; CLEAR LOPCNT
1520	005604	000137	006014			JMP	EXRAX		; GET NEW NUMBER
1521	005610	005737	001200		1\$:	TST	LOPCNT		; ANY ERRORS?
1522	005614	001407				BEG	6\$; NO
1523	005616	104402	000616			TYPE	, RECOV		
1524	005622	013746	001200			MOV	LOPCNT, -(6)		; GET NUMBER
1525	005626	104406				TYPES			; TYPE IT
1526	005630	104402	000636			TYPE	, CRLF		
1527	005634	104400			6\$:	SCOPE			
1528	005636	052737	000003	001126		BIS	#3, FLAG		; SET COUNTER

N03

1529	005644	104400				SCOPE			
1530	005646	005037	001200			CLR	LOPCNT		: CLEAR COUNTER
1531	005652	004537	011306			JSR	RS, CLEAR		: CLEAR BUFFER
1532	005656	104414			RREAD:	ERCLR			: CLEAR RS REG IF ERRORS
1533	005660	012737	000171	001172		MOV	#171, CMD		: READ WITH I/E
1534	005666	104416				DKCMD			: READ
1535	005670	004737	011770			JSR	PC, WATT		: WAIT FOR INTERRUPT
1536	005674	032777	010000	173124		BIT	#BIT12, ASWR		: COMPARE ?
1537	005702	001003				BNE	RWRED		: NO
1538	005704	004537	010452			JSR	RS, COMPARE		: YES
1539	005710	000400				BR	RWRED		: CONT
1540	005712	032737	001000	001126	RWRED:	BIT	#BIT9, FLAG		: IS THERE AN ERROR?
1541	005720	001435				BEQ	EXRAX		: NO
1542	005722	104432				LOGR			: LOG READ ERR
1543	005724	032777	000100	173074	1\$:	BIT	#BIT6, ASWR		: TYPE ALL ERRORS?
1544	005732	001016				BNE	2\$: YES
1545	005734	032777	001000	173064		BIT	#BIT9, ASWR		: LOOP ON ERROR?
1546	005742	001012				BNE	2\$: YES
1547	005744	005737	001200			TST	LOPCNT		: FIRST ERROR?
1548	005750	001010				BNE	3\$: NO
1549	005752	004737	014224			JSR	PC, PRNT		: TIMEOUT?
1550	005756	001004				BNE	2\$: NO
1551	005760	104402	000674			TYPE	, RANDM		
1552	005764	104402	000607			TYPE	, RDERR		
1553	005770	104006			2\$:	HLT	!DB!DA		
1554	005772	104400			3\$:	SCOPE			
1555	005774	005237	001200			INC	LOPCNT		: UPDATE COUNTER
1556	006000	022737	000010	001200		CMP	#10, LOPCNT		: LAST TRY YET?
1557	006006	001323				BNE	RREAD		: RETRY
1558	006010	004737	006524			JSR	PC, NOREC		: TYPE UNRECOVERABLE
1559	006014	005737	001200		EXRAX:	TST	LOPCNT		: ANY ERRORS?
1560	006020	001402				BEQ	EXRXX		: NO
1561	006022	004737	012012			JSR	PC, TYPREC		: TYPE RECOVERED
1562	006026	104400			EXRXX:	SCOPE			
1563	006030	005237	001156			INC	PASSC		: +1 PASS COUNT
1564	006034	001402				BEQ	1\$: IS TEST DONE?
1565	006036	000137	005172			JMP	WRLG1		: NO
1566	006042	005037	001136		1\$:	CLR	PATNU		: END OF TEST
1567	006046	042737	000400	001126		BIC	#BIT8, FLAG		: CLEAR TEST FLAG
1568	006054	032737	100000	001216		BIT	#BIT15, FLAG3		: LOOP ON THIS TEST?
1569	006062	001402				BEQ	.+6		: NO
1570	006064	000137	005062			JMP	RANEX		: YES

1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622

006070 104400
006072 005037 001134
006076 104426
006100 032737 004000 001126
006106 001404
006110 004737 002212
006114 000137 003246
006120 004737 002262

006124 032737 001000 001126
006132 001404
006134 104426
006136 042737 001000 001126
006144 000002

006146 013777 001134 172666
006154 005037 001176
006160 032737 020000 001122
006166 001021
006170 032737 000004 001126
006176 001415
006200 022737 000042 001136
006206 001411
006210 052777 000010 172616
006216 005077 172614
006222 012777 017436 172610
006230 000435
006232 013777 001140 172600
006240 013702 001130
006244 005402
006246 010277 172564
006252 032737 000400 001126
006260 001033
006262 005737 001212
006266 001416
006270 032737 000040 001122
006276 001403
006300 052737 001000 001172

:CHECK FOR MULTI DISK MODE
:IF IN MULTI DISK MODE REPORT "END"
:IF LAST DISK ON SYSTEM HAS BEEN EXERCISED.

:*****
:TEST 4 TEST FOR MULTI DISK MODE
:*****
TST4: SCOPE
EXTPPR: CLR
 CLR DV :INIT DRIVE
 BIT #BIT11,FLAG :ARE WE IN MULTI DISK MODE
 BEQ EXTPP :NO REPORT "END"
 JSR PC,TRYNX :YES TEST FOR ALL DRIVES
 JMP EXMFLG :RESTART TESTING OF DRIVES
EXTPP: JSR PC,DONEE :GET PASS COUNT

:THIS ROUTINE CLEARS THE DRIVE
:REGISTERS IF THERE WAS AN ERROR
ERCLR: BIT #BIT9,FLAG :ANY ERRORS?
 BEQ 1\$:NO
 CLR DV :CLEAR ALL ERRORS
 BIC #BIT9,FLAG :CLEAR ERROR FLAG
1\$: RTI :EXIT

:ENTER DISK HANDLER BY THE TRAP INSTRUCTION
:ARGUMENT TO TRAP INSTRUCTION IS TWO ORDER
:BYTE OF THE CONTROL REGISTER.
DKCMD: MOV DMA,DRSDA :LOAD DISK ADD
 CLR INTFLG :CLEAR INTERRUPT FLAG
 BIT #BIT13,FLAG2 :IN MAPPING ROUTINE?
 BNE 4\$:YES
 BIT #BIT2,FLAG :MAX DATA TEST?
 BEQ 4\$:NO
 CMP #42,PATNU :RANDOM DATA?
 BEQ 4\$:YES
 BIS #10,DRSCS2 :SET BAI
 CLR DRSWC :64K XFER
 MOV #OUTBUF,DRSBA :SETUP BA
 BR 2\$:CONT
4\$: MOV BUF,DRSBA :LOAD (CMA) BUSS ADDRESS
 MOV WRDCT,R2 :GET NEGATIVE
 NEG R2 :WORD COUNT
 MOV R2,DRSWC :LOAD WC
 BIT #BIT8,FLAG :RANDOM TEST?
 BNE 1\$:YES A PORT ONLY WITH NO MEM MGMT
 TST MMAVA :MEM MGMT AVAILIABLE?
 BEQ 2\$:NO
 BIT #BIT5,FLAG2 :SET A17 IN RSCS1
 BEQ 3\$:NO
 BIS #BIT9,CMD :YES

```

1623 006306 032737 000020 001122 3$: BIT #BIT4,FLAG2 ;SET A16?
1624 006314 001403 BEQ 2$ ;NO
1625 006316 052737 000400 001172 BIT #BIT8,CMD ;YES
1626 006324 032737 000100 001126 2$: BIT #BIT6,FLAG ;MULTI PORT?
1627 006332 001406 BEQ 1$ ;NO
1628 006334 005737 001114 TST AOB1 ;TEST A OR B PORT?
1629 006340 001403 BEQ 1$ ;A PORT
1630 006342 052737 002000 001172 BIS #BIT10,CMD ;B PORT
1631 006350 013777 001172 172454 1$: MOV CMD,ARSCS1 ;LOAD FUNCTION REG.
1632 006356 000002 RTI ;RETURN FROM TRAP
1633
1634 ;RH11 DISK INTERRUPT HANDLER
1635 ;ROUTINE CONTINUES ON ERRORS
1636
1637 006360 042737 001000 001126 DKINT: BIC #BIT9,FLAG ;CLEAR ERROR BIT
1638 006366 005777 172440 TST ARSCS1 ;TEST FOR ERROR
1639 006372 100401 BMI 2$
1640 006374 000425 BR INTEXT ;JUMP IF NO ERRORS
1641 006376 017702 172430 2$: MOV ARSCS1,R2 ;GET CONTENTS OF CS1
1642 006402 042702 037777 BIC #37777,R2 ;CLEAR ALL BUT SC AND TRE
1643 006406 022702 140000 CMP #140000,R2 ;IS SC AND TRE BOTH SET?
1644 006412 001413 BEQ TRUERR ;YES THERE IS SOME KIND OF XFER ERROR
1645 006414 032777 100000 172422 BIT #100000,ARSDS ;IS THE ATA BIT SET?
1646 006422 001007 BNE TRUERR ;YES
1647 006424 104140 HLT !AS!DS ;WRONG UNIT INTERRUPTED
1648 ;IF YOU HAVE JUST POWERED UP A DRIVE OR
1649 ;ARE RUNNING THE POWER FAIL TEST,
1650 ;INTERRUPTS WILL OCCUR FROM DRIVES OTHER
1651 ;THAN THE UNIT UNDER TEST. IF THIS TYPEOUT
1652 ;SHOWS NO ERRORS IN THE REGISTERS OF THE DRIVE
1653 ;UNDER TEST, THAT DRIVE IS OK
1654 006426 012777 177777 172414 1$: MOV #-1,ARSA$ ;CLEAR ALL ATA BITS
1655 006434 013716 001152 MOV HRDR,(SP) ;GET RETURN ADD.
1656 006440 000002 RTI ;RETRY
1657 006442 052737 001000 001126 TRUERR: BIS #BIT9,FLAG ;SET ERROR BIT
1658 006450 032777 004000 172350 INTEXT: BIT #BIT11,ASWR ;HALT ON COMPLETION FLAG
1659 006456 001401 BEQ .+4
1660 006460 000000 HALT ;YES BIT 11 SET IN SWR HALT
1661 006462 032737 002000 001122 BIT #BIT10,FLAG2 ;WAIT IN BACKGROUND TEST?
1662 006470 001402 BEQ 1$ ;NO
1663 006472 012716 012436 2$: MOV #NPRRET,(SP) ;MODIFY RETURN ADD.
1664 006476 010637 001176 1$: MOV SP,INTFLG ;SET INT FLG
1665 006502 000002 RTI ;EXIT
1666
1667 ;ROUTINE TO SET UP TRACK # FROM OPTION
1668 ;ENTER FROM JSR R5, OPDSEL
1669
1670 006504 032737 000040 001126 OPDSEL: BIT #BIT5,FLAG ;OPTIONAL DMA?
1671 006512 001403 BEQ 1$ ;NO
1672 006514 013737 001142 001134 MOV TDMA,DMA ;GET OPT. DMA
1673 006522 000205 1$: RTS R5 ;EXIT

```

```

1674
1675
1676          .EVEN
1677
1678 006524 004737 014224      NOREC: JSR      PC,PRNT      ;TYPEOUT?
1679 006530 001002              BNE      1$          ;NO
1680 006532 104402 000522      TYPE     ,UNKECO
1681 006536 005037 001200      1$: CLR     LOPCNT    ;CLEAR LOOP COUNTER
1682 006542 000207              RTS      PC
1683
1684 006544 052737 040000 001122  FNDTYP: BIS     #BIT14,FLAG2 ;SET CHECK DRIVE TYPE FLAG
1685 006552 004737 002024              JSR     PC,UNTYP   ;CHECK DRIVE TYPE FLAG
1686 006556 042737 040000 001122  BIC     #BIT14,FLAG2 ;CLEAR DRIVE TYPE FLAG
1687 006564 000207              RTS      PC
1688 006566 022737 000004 001166  WHTHU: CMP     #4,RS04DT ;RS03LA?
1689 006574 001004              BNE     1$          ;NO
1690 006576 012737 000040 001130  MOV     #40,WRDCT  ;GET WORD COUNT
1691 006604 000411              BR      2$
1692 006606 012737 000200 001130  1$: MOV     #200,WRDCT ;RS04
1693 006614 005737 001166              TST     RS04DT    ;RS04?
1694 006620 001003              BNE     2$          ;YES
1695 006622 012737 000100 001130  MOV     #100,WRDCT ;NO
1696 006630 000207              RTS      PC
1697
1698

```

E04

CZRSCG RH11-R503-R503 LA-R504 DATA AND RELIABILITY TEST
 CZRSCG.F11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 44
 TST4 TEST FOR MULTI DISK MODE

SEG 0043

```

1699 ;ROUTINE TO CALCULATE VITURAL ADDR
1700
1701 006632 000302 PHYCOV: SWAB R2 ;CALCULATE FROM PHYSICAL ADDR
1702 006634 004737 011756 JSR PC,RRR2
1703 006640 006002 ROR R2
1704 006642 042702 177770 BIC #177770,R2 ;GET REG #
1705 006646 032777 000400 172156 BIT #BIT8,DRSCS1 ;IS A16 SET?
1706 006654 001402 BEQ 1$ ;NO
1707 006656 052702 000010 BIS #BIT3,R2 ;YES
1708 006662 032777 001000 172142 1$: BIT #BIT9,DRSCS1 ;IS A17 SET?
1709 006670 001402 BEQ 2$ ;NO
1710 006672 052702 000020 BIS #BIT4,R2 ;YES
1711 006676 013737 001070 001224 2$: MOV STAMEM,WORK1 ;GET BANK # FOR -A- PORT
1712 006704 005737 001114 TST AOB1 ;ARE WE ON -A- PORT?
1713 006710 001403 BEQ 3$ ;YES
1714 006712 013737 001074 001224 MOV STBCOM,WORK1 ;NO -B- PORT
1715 006720 163702 001224 3$: SUB WORK1,R2 ;GET STARTING BANK #
1716 006724 062702 000001 ADD #1,R2 ;GET OFFSET FOR REG #
1717 006730 000302 SWAB R2 ;GET BANK # INTO
1718 006732 006102 ROL R2 ;UPPER BITS
1719 006734 006102 ROL R2
1720 006736 006102 ROL R2
1721 006740 006102 ROL R2
1722 006742 006102 ROL R2
1723 006744 017737 172070 001224 MOV DRSCBA,WORK1 ;GET OFFSET FOR ADDR IF ANY
1724 006752 162737 000002 001224 SUB #2,WORK1 ;CORRECT IT
1725 006760 042737 160000 001224 BIC #160000,WORK1 ;CLEAR JUNK
1726 006766 050237 001224 BIS R2,WORK1 ;GET REG NO
1727 006772 013702 001224 MOV WORK1,R2
1728 006776 000207 RTS PC
1729
1730 007000 012777 006360 172054 VECTRR: MOV #DKIN1,DRSVEC ;SETUP INTERRUPT VECTORS
1731 007006 013737 001066 177776 MOV PRIORITY,PS ;PRIORITY 4
1732 007014 000207 RTS PC
1733
1734 ;THIS ROUTINE IS USED FOR DELAYING THE START OF THIS PROGRAM
1735 ;IF POWER FAILED DURING TESTING. THIS WILL GIVE THE DRIVES TIME TO GET UP
1736 ;TO SPEED. THE DELAY WILL BE ABOUT 3-5 MINUTES DEPENDING UPON THE PROCESSOR
1737
1738 007016 012737 000677 001222 TIMUP: MOV #677,WORK
1739 007024 012737 177777 001224 1$: MOV #177777,WORK1
1740 007032 000240 2$: NOP
1741 007034 005337 001224 DEC WORK1
1742 007040 001374 BNE 2$
1743 007042 005337 001222 DEC WORK
1744 007046 001366 BNE 1$
1745 007050 000137 003232 JMP ADTST
  
```

```

1746          ;ROUTINE TO SETUP DISK BUFFERS
1747          ;ADD WORD COUNT TO STARTING DISK ADDRESSES
1748          ;COMPARE CALCULATED ADDRESS TO TERMINATING ADDRESS
1749
1750 007054 032737 000040 001126 DISBUF: BIT      #BITS,FLAG      ;DID OPERATOR SELECT PATTERNS
1751 007062 001402          BEQ      2$          ;NO
1752 007064 000137 007346          JMP      BUFEXIT      ;YES
1753 007070 022737 000042 001136 2$:  CMP      #42,PATNU    ;RANDOM PATTERN?
1754 007076 001443          BEQ      1$          ;YES
1755 007100 032737 000004 001126          BIT      #BIT2,FLAG    ;MAX TST?
1756 007106 001437          BEQ      1$          ;NO
1757 007110 022737 000004 001166          CMP      #4,RS04DT    ;RS03LA?
1758 007116 001010          BNE      4$          ;NO
1759 007120 022737 004000 001134          CMP      #4000,DMA    ;DONE YET?
1760 007126 001507          BEQ      BUFEXIT      ;YES
1761 007130 062737 004000 001134          ADD      #4000,DMA    ;UPDATE DMA
1762 007136 000207          RTS      PC
1763 007140 005737 001166          4$:  TST      RS04DT      ;RS04?
1764 007144 001010          BNE      3$          ;YES
1765 007146 022737 006000 001134          CMP      #6000,DMA    ;RS03
1766 007154 001474          BEQ      BUFEXIT      ;DONE GET OUT
1767 007156 062737 002000 001134          ADD      #2000,DMA    ;UPDATE DMA
1768 007164 000207          RTS      PC
1769 007166 022737 007000 001134 3$:  CMP      #7000,DMA    ;DONE YET?
1770 007174 001464          BEQ      BUFEXIT      ;YES
1771 007176 062737 001000 001134          ADD      #1000,DMA    ;UPDATE ADDR
1772 007204 000207          RTS      PC
1773 007206 004737 007562          1$:  JSR      PC,BLSZ      ;DEFINE BLOCK SIZE
1774 007212 013737 001154 001224          MOV      BLOCK,WORK1
1775 007220 005237 001134          INCSEC: INC      DMA      ;+1 SECTOR COUNT
1776 007224 022737 010000 001134          CMP      #10000,DMA   ;DONE YET?
1777 007232 001445          BEQ      BUFEXIT      ;YES
1778 007234 005337 001154          DEC      BLOCK      ;-1 FROM BLOCK COUNT
1779 007240 001401          BEQ      COMDAR      ;CMP DMA TO RSDA
1780 007242 000766          BR       INCSEC      ;RECYCLE
1781 007244 032737 001000 001126 COMDAR: BIT      #BIT9,FLAG    ;ANY ERRORS?
1782 007252 001401          BEQ      1$          ;NO ERRORS DO COMPARE ON RSDA
1783 007254 000207          RTS      PC          ;ERRORS DO NOT COMPARE RSDA
1784 007256 023777 001134 171556 1$:  CMP      DMA,RSDA     ;COMPARE RSDA WITH DMA
1785 007264 001425          BEQ      CMDAE       ;SHOULD BE EQUAL
1786 007266 104432          LOGR          ;AFTER TRANSFER RSDA AND DMA SHOULD BE =
1787          ;IF NOT, RSDA IS NOT CORRECT. DMA CONTAINS
1788          ;WHAT RSDA SHOULD =
1789 007270 013701 001134          MOV      DMA,GOOD    ;GET DMA FOR CORRECT ANS IN GOOD
1790 007274 017700 171542          MOV      RSDA,BAD    ;GET RSDA INTO BAD
1791 007300 104000          HLT          ;RSDA=BAD DMA=GOOD SEE COMMENTS 7 LINES ABOVE
1792 007302 004737 014224          JSR      PC,PRNT     ;TYPEOUT?
1793 007306 001014          BNE      CMDAE       ;NO
1794 007310 011637 001222          MOV      (SP),WORK   ;GET TEST PC FROM WHERE IT CAME
1795 007314 104402 007320          TYPE      +2        ;ASCIZ " TST PC="
1796 007332 013746 001222          MOV      WORK,-(E)   ;PUT WORK ON STACK
1797 007336 104406          TYPES          ;TYPE STACK IN OCTAL - SUPPRESS
1798 007340 105737 001126          CMDAE: TSTB      FLAG  ;LAST DISK BUFFER?
1799 007344 100032          BPL      BUFINX     ;NO
  
```

1800	007346	005037	001134		BUFEXIT: CLR	DMA	: CLEAR ADDRESS BITS LAST DISK BUFFER
1801	007352	062716	000002		ADD	#2 (6)	: INC STOCK POINTER
1802	007356	042737	000200	001126	AKH: BIC	#200, FLAG	: CLEAR LAST DISK BUFFER FLAG
1803	007364	032737	000400	001126	BIT	#BIT8, FLAG	: RANDOM TEST OR ADDR TEST?
1804	007372	001404			BEQ	1\$: NO
1805	007374	013737	001144	001130	2\$: MOV	SWRDCT, WRDCT	
1806	007402	000466			BR	EXTLR	: EXIT
1807	007404	032737	000100	001126	1\$: BIT	#BIT6, FLAG	: MULTI PORT?
1808	007412	001770			BEQ	2\$: NO
1809	007414	005737	001114		TST	AOB1	: A OR B PORT?
1810	007420	001765			BEQ	2\$: A PORT
1811	007422	013737	001112	001130	MOV	WDCTB, WRDCT	: B PORT
1812	007430	000453			BR	EXTDR	: GET OUT
1813	007432	005037	001226		BUFINX: CLR	WORK2	: CLEAR WORK2 FOR BLOCK COUNTER
1814	007436	013702	001134		MOV	DMA, R2	: PUT WORKING DISK ADD INTO WORK
1815	007442	005237	001226		XINCSEC: INC	WORK2	: INCREMENT BLOCK COUNT
1816	007446	022702	007777		CMP	#7777, R2	: CMP FOR LAST SECTOR
1817	007452	001405			BEQ	XINCSUR	: +1 SURFACE LAST SECTOR BRANCH
1818	007454	005202			INC	R2	: INC DMA
1819	007456	005337	001224		DEC	WORK1	: DEC BLOCK COUNT
1820	007462	001367			BNE	XINCSEC	: FILLED STANDARD BUFFER YET?
1821	007464	000734			BR	AKH	: WILL TAKE STANDARD SIZE WORD COUNT
1822	007466	013737	001226	001130	XINCSUR: MOV	WORK2, WRDCT	: SETTING UP BLOCK COUNT
1823	007474	000241			CLC		: FOR NON STANDARD BUFFER SIZE
1824	007476	006137	001130		ROL	WRDCT	
1825	007502	006137	001130		ROL	WRDCT	
1826	007506	006137	001130		ROL	WRDCT	
1827	007512	006137	001130		ROL	WRDCT	
1828	007516	006137	001130		ROL	WRDCT	
1829	007522	022737	000004	001166	CMP	#4, R504DT	: R503LA?
1830	007530	001410			BEQ	1\$: YES
1831	007532	000241			CLC		
1832	007534	006137	001130		ROL	WRDCT	
1833	007540	005737	001166		TST	R504DT	: R504?
1834	007544	001402			BEQ	1\$: NO
1835	007546	006137	001130		ROL	WRDCT	: YES
1836	007552	052737	000200	001126	1\$: BIS	#200, FLAG	: SET LAST DISK BUFFER FLAG
1837	007560	000207			EXTDR: RTS	PC	: EXIT

H04

```

1838 ;THIS ROUTINE CONVERTS A WORD COUNT TO A BLOCK COUNT
1839 007562 022737 000004 001166 BLSZ:  CMP    #4,RS04DT    ;RS03LA
1840 007570 001004                BNE    3$                ;NO
1841 007572 012737 000037 001154      MOV    #37,BLOCK        ;YES
1842 007600 000411                BR     2$                ;CONTINUE
1843 007602 012737 000177 001154 3$:   MOV    #177,BLOCK       ;SETUP FOR R504
1844 007610 005737 001166                TST    RS04DT           ;RS04?
1845 007614 001003                BNE    2$                ;YES
1846 007616 012737 000077 001154 1$:   MOV    #77,BLOCK        ;PUT SECTOR SIZE INTO BLOCK
1847 007624 013702 001130 2$:   MOV    WRDCT,R2         ;FETCH WORD COUNT
1848 007630 033702 001154                BIT    BLOCK,R2         ;ARE THEY EQUAL?
1849 007634 001406                BEQ    RORBLK           ;YES
1850 007636 043702 001154                BIC    BLOCK,R2         ;SET UP BLOCK OVERFLOW
1851 007642 005237 001154                INC    BLOCK
1852 007646 063702 001154                ADD    BLOCK,R2
1853 007652 000241      RORBLK: CLC
1854 007654 006002                ROR    R2
1855 007656 022737 000004 001166      CMP    #4,RS04DT        ;RS03LA
1856 007664 001003                BNE    2$                ;NO
1857 007666 004737 011756                JSR    PC,RRR2          ;YES
1858 007672 000410                BR     1$                ;CONTINUE
1859 007674 000241      2$:   CLC
1860 007676 006002                ROR    R2
1861 007700 004737 011756                JSR    PC,RRR2
1862 007704 005737 001166                TST    RS04DT          ;RS04?
1863 007710 001401                BEQ    1$                ;NO
1864 007712 006002                ROR    R2                ;YES
1865 007714 010237 001154 1$:   MOV    R2,BLOCK         ;BLOCK COUNT
1866 007720 000207                RTS    PC                ;EXIT
1867
1868 ;ROUTINE TO SELECT DATA PATTERNS FOR TEST
1869 ;ENTER FROM JSR R5,PASEL
1870 007722 012737 010356 000004 PASEL: MOV    #MEM,2#4        ;SETUP TRAP
1871 007730 012737 000340 000006      MOV    #340,2#6        ;VECTOR
1872 007736 013700 001136                MOV    PATNU,R0        ;SET UP PATTERN NUMBER
1873 007742 010003                MOV    R0,R3           ;GET PATTERN #
1874 007744 000241                CLC                    ;MAKE IT =
1875 007746 006003                ROR    R3              ;TO PATTERN # IN LISTING
1876 007750 010377 171054                MOV    R3,2DISPLAY     ;DISPLAY PATTERN #
1877 007754 013737 001130 001222      MOV    WRDCT,WORK      ;SET UP WORK
1878 007762 013701 001116                MOV    VADDR,R1        ;LOC. OF OUTBUFFER
1879 007766 022700 000042 1$:   CMP    #42,R0          ;TEST FOR RANDOM DATA NUMBER
1880 007772 001424                BEQ    RANDOM           ;GO GENERATE RANDOM DATA
1881 007774 032737 000004 001126      BIT    #BIT2,FLAG      ;MAX TST?
1882 010002 001404                BEQ    2$                ;NO
1883 010004 016037 000254 017436      MOV    PATO(0),OUTBUF  ;GET PATTERN
1884 010012 000205                RTS    R5
1885 010014 016000 000254 2$:   MOV    PATO(0),R0
1886 010020 010021      FILDAT: MOV    R0,(1)+        ;FILL BUFFER
1887 010022 005337 001222                DEC    WORK            ;DEC. WORK COUNT
1888 010026 001374                BNE    FILDAT          ;LOAD NEXT WORD
1889 010030 012737 000006 000004 PASEX: MOV    #6,2#4        ;RESTORE
1890 010036 005037 000006                CLR    2#6             ;TRAP
1891 010042 000205                RTS    R5                ;BUFFER FULL

```

```

;RANDOM DATA GENERATOR SUBROUTINE
1892
1893
1894 010044 013737 010210 010214 RANDOM: MOV LONUM,LCSAV
1895 010052 013737 010212 010216 MOV HINUM,HISAV
1896 010060 013700 010210 RAND1: MOV LONUM,R0 ;SET UP R0 WITH 5 DIGITS LOW
1897 010064 013704 010212 MOV HINUM,R4 ;SET UP R1 WITH 5 DIGITS HIGH
1898 010070 012703 000007 MOV #7,R3 ;SET UP SHIFT COUNT
1899 010074 005002 CLR R2 ;CLEAR R2
1900 010076 006300 SHIFT: ASL R0 ;SHIFT R0 LEFT AND
1901 010100 006104 ROL R4 ;ROTATE CARRY INTO LSB OF R1 INTO
1902 010102 006102 ROL R2 ;ROTATE CARRY OUT OF R1 INTO R2
1903 010104 005303 DEC R3 ;DECREMENT R3
1904 010106 001373 BNE SHIFT ;CONTINUE SHIFT LOOP
1905 010110 063700 010210 ADD LONUM,R0 ;ADDN IN NUMBER TO MAKE X 129
1906 010114 005504 ADC R4 ;PROPOGATE CARRY
1907 010116 063704 010212 ADD HINUM,R4 ;ADDN IN NUMBER TO MAKE X 129
1908 010122 005502 R2 ;PROPOGATE CARRY
1909 010124 062700 001057 ADD #1057,R0 ;ADDN LOW CONSTANT
1910 010130 005504 ADC R4 ;PROPOGATE CARRIES
1911 010132 005502 ADC R2 ;PROPOGATE AGAIN
1912 010134 062704 047401 ADD #47401,R4 ;ADDN HIGH CONSTANT
1913 010140 005502 R2 ;PROPOGATE CARRY
1914 010142 062702 000006 ADD #6,R2 ;ADDN HIGHEST CONSTANT
1915 010146 062700 000002 ADD #2,R0 ;REPRIME R0 WITH HIGH DIGIT
1916 010152 005504 ADC R4 ;PROPOGATE CARRY
1917 010154 010037 010210 MOV R0,LONUM ;PUT R0 BACK IN LONUM
1918 010160 010021 MOV R0,(1)+ ;HOLD LONUM FOR PROGRAM
1919 010162 005337 001222 DEC WORK
1920 010166 001406 BEQ EXGEN
1921 010170 010437 010212 MOV R4,HINUM ;PUT R1 BACK IN HINUM
1922 010174 010421 MOV R4,(1)+ ;HOLD HINUM FOR PROGRAM
1923 010176 005337 001222 DEC WORK
1924 010202 001326 BNE RAND1
1925 010204 000137 010030 EXGEN: JMP PASEX ;RETURN TO PROGRAM
1926 010210 000000 LONUM: 0
1927 010212 000000 HINUM: 0
1928 010214 000000 LOSAV: 0
1929 010216 000000 HISAV: 0
1930
1931 010220 013737 001214 001144 RESTOR: MOV SAVWC,SWRDCT ;RESTORE ORIGINAL
1932 010226 013737 001144 001130 MOV SWRDCT,WRDCT ;WORD COUNT
1933 010234 013737 001220 001112 MOV SAVWCB,WDCB
1934 010242 000205 RTS R5
    
```



```

1935      :RANDOM DATA GENERATOR SUBROUTINE
1936      :WHEN SWITCH = 0 WE COME HERE
1937
1938 010244 013700 010352  RAND:  MOV      LONUM1,R0      ;SET UP R0 WITH 5 DIGITS LOW
1939 010250 013704 010354  MOV      HINUM1,R4      ;SET UP R1 WITH 5 DIGITS HIGH
1940 010254 012703 000007  MOV      #7,R3         ;SET UP SHIFT COUNT
1941 010260 005002  CLR      R2           ;CLEAR R2
1942 010262 006300  SHIFT1: ASL     R0       ;SHIFT R0 LEFT AND
1943 010264 006104  ROL     R4           ;ROTATE CARRY INTO LSB OF R1 INTO
1944 010266 006102  ROL     R2           ;ROTATE CARRY OUT OF R1 INTO R2
1945 010270 005303  DEC     R3           ;DECREMENT R3
1946 010272 001373  BNE    SHIFT1       ;CONTINUE SHIFT LOOP
1947 010274 063700 010352  ADD     LONUM1,R0     ;ADDN IN NUMBER TO MAKE X 129
1948 010300 005504  ADC     R4           ;PROPOGATE CARRY
1949 010302 063704 010354  ADD     HINUM1,R4     ;ADDN IN NUMBER TO MAKE X 129
1950 010306 005502  ADC     R2           ;PROPOGATE CARRY
1951 010310 062700 001057  ADD     #1057,R0      ;ADDN LOW CONSTANT
1952 010314 005504  ADC     R4           ;PROPOGATE CARRIES
1953 010316 005502  ADC     R2           ;PROPOGATE AGAIN
1954 010320 062704 047401  ADD     #47401,R4     ;ADDN HIGH CONSTANT
1955 010324 005502  ADC     R2           ;PROPOGATE CARRY
1956 010326 062702 000006  ADD     #6,R2         ;ADDN HIGHEST CONSTANT
1957 010332 062700 000002  ADD     #2,R0         ;REPRIME R0 WITH HIGH DIGIT
1958 010336 005504  ADC     R4           ;PROPOGATE CARRY
1959 010340 010037 010352  MOV     R0,LONUM1     ;PUT R0 BACK IN LONUM
1960 010344 010437 010354  MOV     R4,HINUM1     ;PUT R1 BACK IN HINUM
1961 010350 000205  EXGEN1: RTS          ;RETURN TO PROGRAM
1962 010352 000000  LONUM1: 0
1963 010354 000000  HINUM1: 0
1964
1965      ;TRAP OUT ROUTINE WHEN CREATING DATA BUFFER
1966
1967 010356  MEM:
1968 010356 104402 010362  TYPE   .,+2          ;.ASCIZ <15><12>"NXM PORT "
1969 010376 005737 001114  TST   AOB1          ;FIND WHAT DATA BUFFER
1970 010402 001004  BNE   3$           ;BRANCH IF B
1971 010404 104402 010410  TYPE   .,+2          ;.ASCIZ "A"
1972 010412 000403  BR    4$
1973 010414 3$:
1974 010414 104402 010420  TYPE   .,+2          ;.ASCIZ "B"
1975 010422 012737 000006 000004 4$:  MOV   #6,2#4        ;RESTORE
1976 010430 005037 000006  CLR   2#6          ;TRAP
1977 010434 032777 100000 170364  BIT   #BIT15,2SWR  ;HALT?
1978 010442 001401  BEQ   2$           ;NO
1979 010444 000000  HALT
1980 010446 000137 001234 2$:  JMP   2#BEGIN
    
```

K04

```

1981                                     ; THIS ROUTINE COMPARES THE DATA READ AGAINST THE DATA EXPECTED.
1982                                     ; ALL ERRORS ARE REPORTED TO THE OPERATOR. IF BIT 4 OF THE SWITCH
1983                                     ; REGISTER IS SET, THIS ROUTINE WILL CONTINUE COMPARING AFTER AN ERROR HAS BEEN
1984                                     ; FOUND AND WILL REPORT UP TO 8 VERIFY ERRORS WITHIN THE SAME INPUT OPERATION.
1985
1986 010452 012737 177770 001146 COMPAR: MOV      #-10, ERCOUNT      ; ERROR RETRY COUNTER
1987 010460 052737 000010 001122          BIS      #BIT3, FLAG2      ; DOING COMPARE
1988 010466 013737 001130 001226          MOV      WRDCT, WORK2      ; GET THE WORD COUNT
1989 010474 013737 001116 001150          MOV      VADDR, SAVE      ; SET UP OUTBUFFER POINTER
1990 010502 005037 001174                   CLR      SWITCH           ; CLEAR RANDOM PATTERN FLAG
1991 010506 013737 010214 010352          MOV      LOSAV, LONUM1     ; GET RANDOM BASE NOS.
1992 010514 013737 010216 010354          MOV      HISAV, HINUM1
1993 010522 005737 001136                   TST      PATNU            ; TEST FOR PATTERN 0
1994 010526 001015                   BNE      1$              ; NO
1995 010530 005037 001222                   CLR      WORK            ; CLEAR COUNTER
1996 010534 062737 000001 001222 2$:      ADD      #1, WORK         ; INC COUNTER
1997 010542 001003                   BNE      3$              ; INTERRUPT YET?
1998 010544 104054                   HLT      !DA!WC!DS       ; TIMED OUT NOT INTERRUPT
1999 010546 000137 001234                   JMP      @#BEGIN
2000 010552 005737 001176                   TST      INTFLG          ; TEST FOR INT
2001 010556 001766                   BEQ      2$              ; WAIT FOR INT BEFORE COMPARING
2002 010560 000426                   BR       CMPLP1          ; CONT
2003 010562 022737 000042 001136 1$:      CMP      #42, PATNU      ; IS THIS RANDOM PATTERN?
2004 010570 001022                   BNE      CMPLP1          ; BRANCH IF YES
2005 010572 005737 001176                   CMPLP:  TST      INTFLG   ; INTERRUPT YET?
2006 010576 001775                   BEQ      CMPLP           ; NO WAIT
2007 010600 005737 001174                   TST      SWITCH
2008 010604 001007                   BNE      2$              ;
2009 010606 004537 010244                   JSR      RS, RAND
2010 010612 013701 010352                   MOV      LONUM1, GOOD    ; GET EVEN RANDOM WORD
2011 010616 010637 001174                   MOV      SP, SWITCH      ; SET RANDOM PATTERN FLAG
2012 010622 000411                   BR       WRDCMP
2013 010624 005037 001174                   CLR      SWITCH         ;
2014 010630 013701 010354                   MOV      HINUM1, GOOD
2015 010634 000404                   BR       WRDCMP
2016 010636 013700 001136                   CMPLP1: MOV      PATNU, R0
2017 010642 016001 000254                   MOV      PATD(R0), GOOD
2018 010646 160177 170276                   WRDCMP: SUB      GOOD, @SAVE ; COMPARE DATA
2019 010652 001017                   BNE      WDERA           ; WORD IN ERROR
2020 010654 005337 001226                   WRDINC: DEC      WORK2    ; DECREMENT THE WORD COUNT
2021 010660 001410                   BEQ      ADAM           ; EXIT ROUTINE IF ZERO
2022 010662 062737 000002 001150          ADD      #2, SAVE        ; UPDATE PATTERN ADDRESS
2023 010670 022737 000042 001136          CMP      #42, PATNU     ; IS THIS RANDOM PATTERN
2024 010676 001735                   BEQ      CMPLP           ; BRANCH IF YES
2025 010700 000762                   BR       WRDCMP         ; COMPARE NEXT WORD
2026 010702 042737 000010 001122  ADAM:  BIC      #BIT3, FLAG2    ; DONE WITH COMPARE
2027 010710 000205                   RTS                      ; EXIT THIS ROUTINE

```

TST4 TEST FOR MULTI DISK MODE

2028	010712	005737	001176		WDERR:	TST	INTFLG	:DID INTERRUPT OCCUR YET?
2029	010716	001753				BEQ	WRDCMP	:BRANCH IF NO
2030	010720	032777	000100	170100		BIT	#BIT6,ASWR	:TRY ALL?
2031	010726	001006				BNE	10\$:YES
2032	010730	005737	001200			TST	LOPCNT	:FIRST READ ERROR?
2033	010734	001403				BEQ	10\$:YES
2034	010736	005777	170070			TST	ASLS1	:ANY ERRORS?
2035	010742	100757				BMI	ADAM	:YES DO NOT COMPARE
2036	010744	060177	170200		10\$:	ADD	GOOD,ASAVE	
2037	010750	017700	170174			MOV	ASAVE,BAD	:GET GOOD DATA
2038	010754	104436				LOGC		:LOG COMPARE ERROR
2039	010756	032777	001000	170042		BIT	#BIT9,ASWR	:LOOP ON ERROR?
2040	010764	001401				BEQ	11\$:NO
2041	010766	005726				TST	(6)+	:YES UPDATE SP
2042	010770	004737	014224		11\$:	JSR	PC,PRNT	:TYPEOUT?
2043	010774	001007				BNE	3\$:NO
2044	010776	104402	011002			TYPE	..+2	:ASCIZ <15><12>"CMP ERR"
2045	011014	104000			3\$:	HLT		:DATA COMPARE ERROR
2046	011016	004737	014224			JSR	PC,PRNT	:HAD TO DO IT THIS WAY SO
2047	011022	001022				BNE	13\$:PROGRAM COULD LOOP ON ERROR
2048	011024	104402	011030			TYPE	..+2	:ASCIZ " ADDR="
2049	011040	005737	001212			TST	MMAVA	:IS MEM MGMT ON?
2050	011044	001406				BEQ	12\$:NO
2051	011046	013746	177776			MOV	PS, -(6)	:GET PS
2052	011052	013746	001150			MOV	SAVE, -(6)	:GET VIRTUAL ADDR
2053	011056	104412				TYPEA		:CONVERT TO PHY AND TYPE
2054	011060	000403				BR	13\$:CONT
2055	011062	013746	001150		12\$:	MOV	SAVE, -(6)	:GET ADDR
2056	011066	104406				TYPES		:TYPE IT
2057	011070	005037	001154		13\$:	CLR	BLOCK	:CLEAR THE BLOCK COUNTER
2058	011074	013702	001130			MOV	WRDCT,R2	:GET THE WORD COUNT
2059	011100	005202				INC	R2	:CORRECT FOR DA CALCULATIONS
2060	011102	163702	001226			SUB	WORK2,R2	:DETERMINE DISTANCE OF FAILURE INTO BUFFER
2061	011106	022737	000004	001166	2\$:	CMP	#4,RS04DT	:RS03LA?
2062	011114	001003				BNE	14\$:NO
2063	011116	162702	000040			SUB	#40,R2	
2064	011122	000410				BR	9\$:CONTINUE
2065	011124	005737	001166		14\$:	TST	RS04DT	:RS04?
2066	011130	001403				BEQ	7\$:NO
2067	011132	162702	000200			SUB	#200,R2	:RS03
2068	011136	000402				BR	9\$:CONT
2069	011140	162702	000100		7\$:	SUB	#100,R2	
2070	011144	100403			9\$:	BMI	8\$	
2071	011146	005237	001154			INC	BLOCK	:UPDATE BLOCK COUNT FOR EACH 400 WORDS
2072	011152	000755				BR	2\$	

M04

2073	011154	022737	000004	001166	8\$:	CMP	#4,R504DT	;R504LA
2074	011162	001003				BNE	15\$;NO
2075	011164	062702	000040			ADD	#40,R2	;RESTORE POSITIVE #
2076	011170	000410				BR	6\$;CONTINUE
2077	011172	005737	001166		15\$:	TST	R504DT	;R504?
2078	011176	001403				BEQ	4\$;NO
2079	011200	062702	000200			ADD	#200,R2	;R504
2080	011204	000402				BR	6\$;CONT
2081	011206	062702	000100		4\$:	ADD	#100,R2	;RESTORE POSITIVE NUMBER
2082	011212	013737	001134	001224	6\$:	MOV	DMA,WORK1	;GET HEAD AND SECTOR ADDRESS
2083	011220	063737	001154	001224	5\$:	ADD	BLOCK,WORK1	
2084	011226	004737	014224			JSR	PC,PRNT	;TYPEOUT?
2085	011232	001014				BNE	1\$;NO
2086	011234	104402	011240			TYPE	.+2	;ASCIZ "DA="
2087	011246	013746	001224			MOV	WORK1,-(6)	;PUT WORK1 ON STACK
2088	011252	104406				TYPES		;TYPE STACK IN OCTAL - SL.RESS
2089	011254	104402	011260			TYPE	.+2	;ASCIZ <15><12>
2090	011264	032777	000020	167534	1\$:	BIT	#BIT4,JSWR	;RETRY?
2091	011272	001405				BEQ	CLEAR	;NO
2092	011274	005237	001146			INC	ERCOUNT	;UPDATE ERROR COUNTER
2093	011300	001402				BEQ	CLEAR	
2094	011302	000137	010654			JMP	WRDINC	
2095	011306	032737	000004	001126	CLEAR:	BIT	#BIT2,FLAG	;XFER TEST?
2096	011314	001404				BEQ	3\$;NO
2097	011316	032737	010000	001126		BIT	#BIT12,FLAG	;READ?
2098	011324	001412				BEQ	2\$;NO
2099	011326	013700	001116		3\$:	MOV	VADDR,R0	;GET STARTING ADDR OF BUFFER
2100	011332	013701	001130			MOV	WRDCT,R1	;NOW
2101	011336	005020			1\$:	CLR	(R0)+	;CLEAR BUFFER
2102	011340	005301				DEC	R1	;COUNT LOCATIONS
2103	011342	001375				BNE	1\$;WAIT TILL DONE
2104	011344	042737	000010	001122		BIC	#BIT3,FLAG2	;DONE WITH COMPARE
2105	011352	000205			2\$:	RTS	R5	;NOW GET OUT
2106								
2107	011354	013737	001072	017436	APOINT:	MOV	SAVAST,OUTBUF	;SET STARTING ADDR FOR OUTBUF
2108	011362	013737	001072	001116		MOV	SAVAST,VADDR	;SAVE OUTBUF ADDR
2109	011370	005737	001212			TST	MMAVA	;MEM MGMT?
2110	011374	001411				BEQ	EXTT	;NO
2111	011376	013702	001100			MOV	SAVMGA,R2	;SET UP MEM MGMT
2112	011402	004737	012040		MMSET:	JSR	PC,STMM2	;SETUP MEM MGMT
2113	011406	010237	001116			MOV	R2,VADDR	
2114	011412	013737	001120	017436		MOV	PHADDR,OUTBUF	
2115	011420	000207			EXTT:	RTS	PC	
2116								
2117	011422	013737	001112	001130	BPOINT:	MOV	WDCTB,WRDCT	;GET WC FOR B PORT
2118	011430	013737	001076	017436		MOV	SAVCPU,OUTBUF	
2119	011436	013737	001076	001116		MOV	SAVCPU,VADDR	
2120	011444	005737	001212			TST	MMAVA	;MEM MGMT AVAILABLE?
2121	011450	001763				BEQ	EXTT	;NO
2122	011452	013702	001104			MOV	SAVMGC,R2	
2123	011456	000751				BR	MMSET	

```

2124 ;TYPE CAN NOT WRITE BLOCK
2125
2126 011460 004737 014224 WTNO: JSR PC,PRNT ;TYPEOUT?
2127 011464 001001 BNE 1$ ;NO
2128 011466 000000 HALT ;HALT CANT WRITE BLOCK
2129 011470 005037 001200 1$: CLR LOPCNT ;CLEAR ERR COUNTER
2130 011474 000207 RTS PC
2131
2132 ;ROUTINE TO SET UP STARTING ADDRESS FOR ALL PORTS
2133 ;AND TO CREATE WORD COUNT MAX= 20K
2134
2135 011476 013702 001070 EXTMEM: MOV STAMEM,R2 ;GET BANK #
2136 011502 005702 TST R2 ;DID HE TYPE 0?
2137 011504 001001 BNE 3$ ;NO
2138 011506 005202 INC R2 ;YES MAKE 1
2139 011510 005737 001212 3$: TST MAVA ;BRANCH IF MEM MGMT AVAILABLE
2140 011514 001021 BNE 1$
2141 011516 000241 CLC
2142 011520 004737 011756 JSR PC,RRR2
2143 011524 010237 001072 MOV R2,SAVAST ;SAVE A STARTING ADDR
2144 011530 032737 000100 001126 BIT #BIT6,FLAG ;IS THERA A B PORT?
2145 011536 001430 BEQ 2$ ;NO
2146 011540 013702 001074 MOV STBCOM,R2 ;YES GET STARTING
2147 011544 000241 CLC
2148 011546 004737 011756 JSR PC,RRR2
2149 011552 010237 001076 MOV R2,SAVCPU ;SAVE IT
2150 011556 000420 BR 2$ ;GET WC
2151 011560 000302 1$: SWAB R2
2152 011562 006002 ROR R2
2153 011564 010237 001100 MOV R2,SAVMGA ;SAVE ADDR FOR A PORT
2154 011570 032737 000100 001126 BIT #BIT6,FLAG ;IS THERE B PORT?
2155 011576 001410 BEQ 2$ ;NO
2156 011600 013702 001074 MOV STBCOM,R2
2157 011604 000302 SWAB R2
2158 011606 006002 ROR R2
2159 011610 010237 001102 MOV R2,SAVMGB ;SAVE B STARTING ADDR
2160 011614 010237 001104 MOV R2,SAVMGC ;SAVE CPU STARTING ADDR
2161 011620 013702 001106 2$: MOV SIZEAP,R2 ;GET 4K BLOCK COUNT
2162 011624 005202 INC R2
2163 011626 013703 001070 MOV STAMEM, R3
2164 011632 160302 SUB R3, R2
2165 011634 022702 000007 CMP #7,R2 ;IS IT GREATER THEN 20K?
2166 011640 101411 BLOS 4$ ;YES MAKE IT 20K
    
```

BOS

2167	011642	000241		8\$:	CLC			
2168	011644	006002			ROR	R2	:NO CONVERT TO WC	
2169	011646	004737	011756		JSR	PC,RRR2		
2170	011652	042702	000077		BIC	#77,R2	:CLEAR BLOCK COUNT	
2171	011656	010237	001144		MOV	R2,SWRDCT	:SAVE -A- PORT WC	
2172	011662	000403			BR	5\$:CONT	
2173	011664	012737	060000	001144	4\$:	MOV	#60000,SWRDC	:MAKE 20K
2174	011672	032737	000100	001126	5\$:	BIT	#BIT6,FLAG	:IS THERE B PORT?
2175	011700	001425			BEG	7\$:NO GET OUT	
2176	011702	013702	001110		MOV	SIZEBP,R2	:GET B 4K COUNT	
2177	011706	005202			INC	R2		
2178	011710	013703	001074		MOV	STBCOM, R3		
2179	011714	160302			SUB	R3, R2		
2180	011716	022702	000007		CMP	#7,R2	:IS IT GREATER THEN 20K?	
2181	011722	101411			BLOS	6\$:YES MAKE 20K	
2182	011724	000241		9\$:	CLC			
2183	011726	006002			ROR	R2	:NO CONVERT TO WC	
2184	011730	004737	011756		JSR	PC,RRR2		
2185	011734	042702	000077		BIC	#77,R2	:CLEAR SECTOR COUNT	
2186	011740	010237	001112		MOV	R2,WDCTB	:SAVE WC FOR -B- PORT	
2187	011744	000403			BR	7\$:GET OUT	
2188	011746	012737	060000	001112	6\$:	MOV	#60000,WDCTB	:MADE 20K WD
2189	011754	000207			7\$:	RTS		
2190								
2191	011756	006002		RRR2:	ROR	R2		
2192	011760	006002			ROR	R2		
2193	011762	006002			ROR	R2		
2194	011764	006002			ROR	R2		
2195	011766	000207			RTS	PC		
2196								
2197	011770	032777	000200	167030	WATT:	BIT	#BIT7,ASWR	:WAIT IN BACKGROUND?
2198	011776	001003			BNE	1\$:NO	
2199	012000	004737	012332		JSR	PC,XWAIT	:YES	
2200	012004	000401			BR	2\$:CONT	
2201	012006	000001		1\$:	WAIT			
2202	012010	000207		2\$:	RTS	PC		
2203								
2204	012012	004737	014224	TYPREC:	JSR	PC,PRNT	:TYPEOUT?	
2205	012016	001007			BNE	1\$:NO	
2206	012020	104402	000616		TYPE	,RECOV		
2207	012024	013746	001200		MOV	LOPCNT,-(6)	:GET COUNT	
2208	012030	104406			TYPES		:TYPE IT	
2209	012032	104402	000636		TYPE	,CRLF		
2210	012036	000207		1\$:	RTS	PC		

C05

```

2211
2212 012040 005737 001212      STMM2: TST      MMAVA      ;MEM MGMT?
2213 012044 001002              BNE      3$      ;YES
2214 012046 000137 012324      JMP      MDON    ;GET OUT
2215 012052 005037 172340      3$: CLR      @#KIPAR0
2216 012056 010237 001150      MOV      R2,SAVE ;SAVE R2
2217 012062 010237 172342      MOV      R2,@#KIPAR1
2218 012066 006302              ASL      R2      ;CALCULATE PHYSICAL ADDR
2219 012070 006302              ASL      R2
2220 012072 006302              ASL      R2
2221 012074 006302              ASL      R2
2222 012076 006302              ASL      R2
2223 012100 042737 000040 001122      BIC      #BITS,FLAG2 ;THIS BIT IS A17
2224 012106 103003              BCC      1$      ;CLEAR A17?
2225 012110 052737 000040 001122      BIS      #BITS,FLAG2 ;SET A17
2226 012116 042737 000020 001122      1$: BIC      #BIT4,FLAG2 ;SET BIT 5 FOR A17
2227 012124 006302              ASL      R2      ;CLEAR A16 FLAG
2228 012126 103003              BCC      2$      ;GET A16 BIT
2229 012130 052737 000020 001122      2$: BIS      #BIT4,FLAG2 ;CLEAR A16
2230 012136 010237 001120      MOV      R2,PHADDR ;SET FLAG FOR A16
2231 012142 013702 001150      MOV      SAVE,R2  ;GET PHYSICAL ADDR
2232 012146 062702 000200      ADD      #200,R2  ;SET UP MEM MGMT
2233 012152 010237 172344      MOV      R2,@#KIPAR2
2234 012156 062702 000200      ADD      #200,R2
2235 012162 010237 172346      MOV      R2,@#KIPAR3
2236 012166 062702 000200      ADD      #200,R2
2237 012172 010237 172350      MOV      R2,@#KIPAR4
2238 012176 062702 000200      ADD      #200,R2
2239 012202 010237 172352      MOV      R2,@#KIPAR5
2240 012206 062702 000200      ADD      #200,R2
2241 012212 010237 172354      MOV      R2,@#KIPAR6
2242 012216 012737 077406 172300      MOV      #200*256.-400+UP+RW,@#KIPDR0 ;SET KIPDR0=RW UP 200 BLOCKS
2243 012224 012737 077406 172302      MOV      #200*256.-400+UP+RW,@#KIPDR1 ;SET KIPDR1=RW UP 200 BLOCKS
2244 012232 012737 077406 172304      MOV      #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
2245 012240 012737 077406 172306      MOV      #200*256.-400+UP+RW,@#KIPDR3 ;SET KIPDR3=RW UP 200 BLOCKS
2246 012246 012737 077406 172310      MOV      #200*256.-400+UP+RW,@#KIPDR4 ;SET KIPDR4=RW UP 200 BLOCKS
2247 012254 012737 077406 172312      MOV      #200*256.-400+UP+RW,@#KIPDR5 ;SET KIPDR5=RW UP 200 BLOCKS
2248 012262 012737 077406 172314      MOV      #200*256.-400+UP+RW,@#KIPDR6 ;SET KIPDR6=RW UP 200 BLOCKS
2249 012270 012737 077406 172316      MOV      #200*256.-400+UP+RW,@#KIPDR7 ;SET KIPDR7=RW UP 200 BLOCKS
2250 012276 012737 007600 172356      MOV      #7600,@#KIPAR7
2251 012304 012702 020000      MOV      #20000,R2
2252 012310 012737 012326 000250      MOV      #MMABTO,@#MMVEC
2253 012316 012737 000001 177572      MOV      #1,@#SR0 ;TURN ON MEM MGMT
2254 012324 000207      MDON: RTS      PC
2255
2256      ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
2257 012326 000000      MMABTO: HALT   ;MEMORY MANAGEMENT TRAP
2258 012330 000002      RTI      ;CAUSED THE ABORT
  
```

```

;BACKGROUND TEST FOR INTERRUPTS
2259
2260
2261 012332 052737 002000 001122 XWAIT: BIS #BIT10,FLAG2 ;WAITING IN BACKGROUND TEST
2262 012340 012737 070000 012446 MOV #70000,NPRCNT ;SETUP TIMEOUT COUNTER
2263 012346 012701 012451 MOV #NPR1+1,R1 ;SETUP WAIT LOOP
2264 012352 112711 000200 MOVB #200,(R1)
2265 012356
2266 012356 105421 2$: NEGB (R1)+
2267 012360 105441 NEGB -(R1)
2268 012362 105421 NEGB (R1)+
2269 012364 105441 NEGB -(R1)
2270 012366 105421 NEGB (R1)+
2271 012370 105441 NEGB -(R1)
2272 012372 105421 NEGB (R1)+
2273 012374 105441 NEGB -(R1)
2274 012376 105421 NEGB (R1)+
2275 012400 105441 NEGB -(R1)
2276 012402 105421 NEGB (R1)+
2277 012404 105441 NEGB -(R1)
2278 012406 105421 NEGB (R1)+
2279 012410 105441 NEGB -(R1)
2280 012412 105421 NEGB (R1)+
2281 012414 105441 NEGB -(R1)
2282 012416 102401 BVS 1$
2283 012420 000000 HALT ;ARITHMETIC OPERATION FAILED RUN DIAG
2284 012422 005337 012446 1$: DEC NPRCNT
2285 012426 001353 BNE 2$
2286 012430 104054 HLT !DA!WC!DS ;TIMED OUT NO INTERRUPT
2287 012432 000137 001234 JMP @#BEGIN
2288 012436 042737 002000 001122 NPRRET: BIC #BIT10,FLAG2 ;CLEAR BKGROUND FLG
2289 012444 000207 RTS PC
2290 012446 000000 NPRCNT: 0
2291 012450 000000 NPR1: 0
2292 ;CLEAR ERROR TABLE
2293
2294 012452 012704 000020 ERRCL: MOV #20,R4 ;CLEAR
2295 012456 012703 017072 MOV #ERTAB,R3 ;ERROR
2296 012462 005023 1$: CLR (R3)+ ;TABLE
2297 012464 005304 DEC R4 ;DONE YET?
2298 012466 001375 BNE 1$ ;NO
2299 012470 005037 001004 CLR PCNT ;CLEAR
2300 012474 005037 001006 CLR PCNT+2 ;PASS COUNT
2301 012500 005037 001124 CLR DROP ;CLEAR ALL DROPPED DRIVES
2302 012504 000205 RTS R5 ;RETURN

```


E05

```

2303 :RH11 POWER FAIL TEST #1
2304 :THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 244. THE PROGRAM
2305 :WRITE THE COMPLETE DISK WITH A 125252 PATTERN. THE PROGRAM WILL THEN
2306 :TELL OPERATOR TO POWER DOWN. UNTIL THE POWER FAIL, THE PROGRAM WILL
2307 :CONTINUE WRITING THE SAME PATTERN ON THE DISK.
2308 :WHEN POWER FAIL OCCURS THE TRANSFER IS ABORTED
2309 :AND THE PROGRAM HALTS. THE OPERATOR SHOULD
2310 :NOW TURN POWER BACK ON. THE PROGRAM RESTARTS AND CHECKS FOR WRITE ERRORS.
2311 :ONLY ONE ERROR IS ACCEPTABLE. THAT ERROR MAY BE AN OPI (BIT13 RSER) OR A DCK
2312 : (BIT 15 RSER). IF THESE ARE THE ONLY ERRORS THAT OCCUR, THE DRIVE IS OK.
2313 :IF NO ERRORS OCCUR, THE PROGRAM WILL TYPE OUT "OK".
2314 :THE PROGRAM WILL THEN TELL YOU WHEN TO POWER DOWN AGAIN
2315
2316 :***ONLY ONE ERROR IS CONSIDERED ACCEPTABLE***
2317 :NOTE: ALL DRIVES ON THE SYSTEM SHOULD BE POWERED OFF EXCEPT
2318 :THE DRIVE UNDER TEST. *****
2319
2320 012506 012706 000500 PFT1: MOV #500,SP ;SET UP STACK
2321 012512 104402 000510 TYPE ,LOADSW
2322 012516 104420 RDOCT
2323 012520 012637 001160 MOV (SP)+,UNNUM
2324 012524 004737 007000 2$: JSR PC,VECTR ;SETUP INT VECTOR
2325 012530 004737 006544 JSR PC,FNDTYP ;TST FOR RS03 OR 04
2326 012534 104426 PFWATT: CLRDV ;CLEAR ALL REG
2327 012536 004737 013312 JSR PC,POWFAL ;WRITE 125252 ON DISK
2328 012542 005037 001134 PFWAT: CLR DMA
2329 012546 012737 012770 000024 MOV #DOWN,24 ;SET UP POWER FAIL VEC.
2330 012554 012737 000340 000026 MOV #340,26
2331 012562 012737 000161 001172 MYBYWR: MOV #161,CMD ;WRITE WITH I/E
2332 012570 104416 DKCMD ;DO IT
2333 012572 004737 011770 JSR PC,WATT ;WAIT FOR INTERRUPT
2334 012576 032737 001000 001126 3$: BIT #BIT9,FLAG ;ANY ERRORS?
2335 012604 001406 IS ;NO
2336 012606 104006 HLT ;DA!DB
2337 012610 012777 177777 166232 MOV #-1,RSAS ;CLEAR ALL
2338 012616 005077 166224 CLR RSER ;ERRORS
2339 012622 004737 007054 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
2340 012626 000755 BR MYBYWR
2341 012630 000744 BR PFWAT

```

F05

```

2342 012632 012737 012640 001152 UPCHK: MOV #1$,HRDR ;RETURN HERE IF WRONG DRIVE INTERRUPTS
2343 012640 005037 001134 1$: CLR DMA
2344 012644 104426 CLRDV ;INIT DRIVE
2345 012646 013737 001066 177776 CHKDAT: MOV PRIORITY,PS
2346 012654 012737 000151 001172 MOV #151,CMD ;WRITECHECK WITH I/E
2347 012662 104416 DKCMD ;DO IT
2348 012664 013737 001066 177776 MOV PRIORITY,PS
2349 012672 004737 011770 JSR PC,WATT ;WAIT FOR INTERRUPT
2350 012676 032737 001000 001126 3$: BIT #BIT9,FLAG ;ANY ERRORS?
2351 012704 001411 BEQ 1$ ;NO
2352 012706 104006 HLT !DB!DA
2353 012710 052737 100000 001122 BIS #BIT15,FLAG2 ;SET ERROR FLAG
2354 012716 005077 166124 CLR #RSEA ;CLEAR ALL
2355 012722 012777 177777 166120 MOV #-1,#RSAS ;ERRORS
2356 012730 004737 007054 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
2357 012734 000744 BR CHKDAT
2358 012736 005737 001122 TST FLAG2 ;ANY ERRORS?
2359 012742 100405 BMI 2$ ;YES
2360 012744 104402 012750 TYPE 1,+2 ;.ASCIZ <15><12>"OK"
2361 012756 042737 100000 001122 2$: BIC #BIT15,FLAG2 ;CLEAR ERROR FLAG
2362 012764 000137 012534 JMP PFWATT ;GO WAIT FOR ANOTHER
2363
2364 ;POWER DOWN ROUTINE - ABORT DISK AND HALT
2365
2366
2367 012770 012737 013000 000024 DOWN: MOV #UPP,24 ;SET POWER FAIL VECTOR
2368 012776 000000 HALT
2369
2370 013000 012737 012770 000024 UPP: MOV #DOWN,24
2371 013006 012706 000500 MOV #500,SP
2372 013012 013777 001160 166014 MOV UNNUM,#RSCS2 ;GET UNIT #
2373 013020 032777 000200 166016 1$: BIT #BIT7,#RSDS ;WAIT FOR DRIVE READY
2374 013026 001774 BEQ 1$
2375 013030 000137 012632 JMP UPCHK ;GO CHECK DISK
  
```

G05

2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422

013034 012706 000500
013040 042737 001000 001122
013046 012737 013070 001152
013054 104426
013056 004737 007000
013062 004737 013312
013066 000401
013070 104426
013072 005037 001134
013076 012737 013242 000024
013104 012737 000340 000026
013112 013737 001066 177776
013120 012737 000151 001172
013126 104416
013130 004737 011770
013134 032737 001000 001126
013142 001411
013144 104002
013146 052737 100000 001122
013154 005077 165666
013160 012777 177777 165662
013166 004737 007054
013172 000747
013174 032737 001000 001122
013202 001733
013204 005737 001122
013210 100405
013212 104402 013216
013224 042737 100000 001122
013232 042737 001000 001122
013240 000710

:POWER FAIL TEST #2
:THIS TEST WILL TEST THE SAME DRIVE THAT WAS TESTED IN THE 1ST POWER FAIL TEST
:THE PROGRAM WILL WRITE THE COMPLETE DISK WITH A 125252 PATTERN AND WILL
:THEN TELL THE OPERATOR TO POWER DOWN THE PROCESSOR.
:THE PROGRAM WILL THEN WRITE CHECK THE DISK WHILE WAITING FOR A POWER FAIL.
:WHEN THE POWER FAIL OCCURS, THE WRITE CHECKING IS ABORTED AND
:THE PROCESSOR WILL HALT.
:THE OPERATOR SHOULD THEN TURN POWER BACK ON, THE PROGRAM WILL
:START WRITE CHECKING THE DISK AGAIN
:***NO ERRORS SHOULD OCCUR.***
:THE PROGRAM WILL TYPE OUT "OK" IF NO ERRORS OCCUR.
:THE PROGRAM WILL THEN TELL YOU TO POWER DOWN.
:DO NOT POWER OFF THE PROCESSOR AGAIN UNTIL THE PROGRAM TELLS YOU SO.
:NOTE: ALL DRIVES ON THE SYSTEM SHOULD BE POWERED OFF
:EXCEPT THE ONE UNDER TEST *****

PFT2: MOV #500,SP ;SET UP STACK
BIC #BIT9,FLAG2 ;CLEAR POWER FAIL
MOV #PWRFL,HRDR ;RETURN HERE IF WRONG DRIVE INT.
CLR DV ;INIT DRIVE
JSR PC,VECTRR ;SETUP INT VECTOR
PWRFL2: JSR PC,POWFAL ;WRITE 125252 ON DISK
BR PWRFL ;WRITE CHECK
PWRFL1: CLR DV ;INIT DRIVE
PWRFL: CLR DMA
MOV #PWRDN,24 ;SET UP POWER FAIL VEC.
MOV #340,26
CHKDSK: MOV PRIORITY,PS ;ENABLE I/E
MOV #151,CMD ;WRITE CHECK WITH I/E
DKCMD ;DO IT
JSR PC,WATT ;WAIT FOR INTERRUPT
3\$: BIT #BIT9,FLAG ;ANY ERRORS?
BEQ 1\$;NO
HLT ;DB ;YES
BIS #BIT15,FLAG2 ;SET ERROR FLAG
CLR ARSER ;CLEAR ALL
MOV #-1,ARSAS ;ERRORS
1\$: JSR PC,DISBUF ;CHECK NEXT BUFFER
BR CHKDSK
BIT #BIT9,FLAG2 ;DID POWER FAIL?
BEQ PWRFL ;NO
TST FLAG2 ;ANY ERRORS?
BMI 2\$;YES
TYPE 1,+2 ;ASCIZ <15><12>"OK"
2\$: BIC #BIT15,FLAG2 ;CLEAR ERRORS
BIC #BIT9,FLAG2 ;CLEAR POWER FAIL FLAG
4\$: BR PWRFL2

H05

```

2423 ;ROUTINE TO ABORT DISK DURING POWER FAIL
2424
2425 013242 012737 013252 000024 PWRDN: MOV #PWRUP,24 ;SET UP RESTART
2426 013250 000000 HALT
2427
2428 013252 012737 013242 000024 PWRUP: MOV #PWRDN,24 ;RESET POWER FAIL VECTOR
2429 013260 012706 000500 MOV #500,SP
2430 013264 013777 001160 165542 MOV UNNUM,DRSCS2 ;GET UNIT #
2431 013272 052737 001000 001122 BIS #BIT9,FLAG2 ;SET POWER FAIL BIT
2432 013300 032777 000200 165536 1$: BIT #BIT7,DRSDS ;WAITING FOR
2433 013306 001774 BEQ 1$ ;DRIVE READY
2434 013310 000667 BR PWRF1 ;GO CHECK DISK
2435
2436
2437 ;ROUTINE TO WRITE THE COMPLETE DISK
2438 ;WITH 125252 PATTERN
2439 ;WRITE CHECK AND REPORT ERRORS IF THEY OCCUR
2440 ;REPORT "OK" AT COMPLETION
2441
2442 013312 012737 000020 001136 POWFAL: MOV #20,PATNU ;SET UP PATTERN
2443 013320 042737 000004 001126 BIC #BIT2,FLAG ;CLEAR XFER MODE FLAG
2444 013326 052737 010000 001122 BIS #BIT12,FLAG2
2445 013334 005037 001134 CLR DMA
2446 013340 012737 020000 017436 MOV #20000,OUTBUF ;GET STARTING ADDR FOR BUF
2447 013346 012737 020000 001116 MOV #20000,VADDR
2448 013354 012737 010000 001144 MOV #10000,SWRDCT ;SETUP WORD COUNT
2449 013362 013737 001144 001130 MOV SWRDCT,WRDCT
2450 013370 005037 001114 CLR AOB1 ;A PORT ONLY
2451 013374 013737 017436 001140 MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
2452 013402 004537 007722 JSR RS,PASEL ;GENERATE DATA BUFFER
2453 013406 012737 000161 001172 WRDNW: MOV #161,CMD ;WRITE WITH I/E
2454 013414 104416 DKCMD ;DO IT
2455 013416 004737 011770 JSR PC,WATT ;WAIT FOR INTERRUPT
2456 013422 012737 000151 001172 2$: MOV #151,CMD ;WRITECHECK I/E
2457 013430 104416 DKCMD ;DO IT
2458 013432 004737 011770 JSR PC,WATT ;WAIT FOR INTERRUPT
2459 013436 032737 001000 001126 4$: BIT #BIT9,FLAG ;ANY ERRORS?
2460 013444 001402 BEQ 1$ ;NO
2461 013446 104006 HLT !DB!DA ;YES
2462 013450 000000 HALT ;CAN NOT WRITE WITHOUT ERROR
2463 013452 004737 007054 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
2464 013456 000753 BR WRDNW ;WRITE NEW BUFFER
2465 013460 104402 TYPE PDOWN
2466 013464 000207 RTS PC
  
```

I05

CZRSCG RH11-R503-R503/LA-R504
CZRSCG.P11 24-JAN-78 07:42

DATA AND RELIABILITY TEST
TST4 TEST FOR MULTI DISK MODE

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 61

SEG 0060

2467	013466	032777	000010	165332	OUT:	BIT	#BIT3,DSWR	:TYPEOUT ERROR COUNT?
2468	013474	001526				BEQ	1\$:NO
2469	013476	005004				CLR	R4	:CLEAR UNIT #
2470	013500	005003				CLR	R3	
2471	013502	053737	001124	001162		BIS	DROP,UNITSV	:RESTORE ALL DRIVES
2472	013510	013737	001162	001222		MOV	UNITSV,WORK	:GET UNITS ON SYSTEM
2473	013516	012705	000401			MOV	#40,R5	:SETUP TEST FOR UNITS
2474	013522	030537	001222		4\$:	BIT	R5,WORK	:IS THIS UNIT ON SYS
2475	013526	001006				BNE	2\$:YES
2476	013530	005204			5\$:	INC	R4	:INC UNIT #
2477	013532	010403				MOV	R4,R3	:SAVE UNIT #
2478	013534	000241				CLC		
2479	013536	006105				ROL	R5	:GET NEXT DRIVE
2480	013540	103501				BCS	3\$:DONE
2481	013542	000767				BR	4\$:FIND NEXT DRIVE
2482	013544	104402	000510		2\$:	TYPE	LOADSW	
2483	013550	010446				MOV	R4,-(6)	:PUT R4 ON STACK
2484	013552	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2485	013554	004737	014032			JSR	PC,GETERR	:GET ERROR COUNT
2486	013560	010304				MOV	R3,R4	:RESTORE UNIT #
2487	013562	104402	013566			TYPE	,+2	:.ASCIZ <15><12>
2488	013572	104402	000564			TYPE	,WRERR	
2489	013576	104402	013602			TYPE	,+2	:.ASCIZ "S "
2490	013606	013746	001202			MOV	WRITER,-(6)	:PUT WRITER ON STACK
2491	013612	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2492	013614	104402	013620			TYPE	,+2	:.ASCIZ <15><12>
2493	013624	104402	000607			TYPE	,RDERR	
2494	013630	104402	013634			TYPE	,+2	:.ASCIZ "S "
2495	013640	013746	001206			MOV	READER,-(6)	:PUT READER ON STACK
2496	013644	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2497	013646	104402	013652			TYPE	,+2	:.ASCIZ <15><12>
2498	013656	104402	000574			TYPE	,WCKERR	
2499	013662	104402	013666			TYPE	,+2	:.ASCIZ "S "
2500	013672	013746	001204			MOV	WCERR,-(6)	:PUT WCERR ON STACK
2501	013676	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2502	013700	104402	013704			TYPE	,+2	:.ASCIZ <15><12>"COMPARE ERRS "
2503	013724	013746	001210			MOV	COMERR,-(6)	:PUT COMERR ON STACK
2504	013730	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2505	013732	104402	013736			TYPE	,+2	:.ASCIZ <15><12>
2506	013742	000672				BR	5\$:GET NEXT DRIVE
2507	013744	043737	001124	001162	3\$:	BIC	DROP,UNITSV	:REDROP DRIVES
2508	013752	062706	000002		1\$:	ADD	#2,SP	:RESTORE SP DUE TO JMP EXIT FROM JSR ROUTINE
2509	013756	005137	001114			COM	A0B1	:SET A OR B PORT FLAG
2510	013762	032777	000040	165036		BIT	#BITS,DSWR	:TYPEOUT PASS COUNT?
2511	013770	001035				BNE	DONE	:NO
2512	013772	104402	013776			TYPE	,+2	:.ASCIZ <15><12>"END PASS "
2513	014012	013746	001006			MOV	PCNT+2,-(6)	:PUT PCNT+2 ON STACK
2514	014016	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2515	014020	104402	014024			TYPE	,+2	:.ASCIZ <15><12>
2516	014030	000415				BR	DONE	

J05

```

2517 014032 006304          GETERR: ASL      R4          ;GET LOC IN
2518 014034 006304          ASL      R4          ;ERR TABLE
2519 014036 062704 017072    ADD      #ERTAB,R4
2520 014042 112437 001202    MOV      (R4)+,WRITER ;GET WRITE ERRS
2521 014046 112437 001206    MOV      (R4)+,READER ;GET READ ERRS
2522 014052 112437 001204    MOV      (R4)+,WCERR  ;GET WRITE CK ERRS
2523 014056 112437 001210    MOV      (R4)+,COMERR ;GET COMPARE ERRS
2524 014062 000207          RTS      PC
2525
2526          .SBTTL          $DONE - BELL AND SCOPE ROUTINE
2527
2528 014064 104400          DONE:  SCOPE          ;TERMINATING SCOPE FOR LOOPING
2529 014066 062737 000001 001006  ADD      #1,PCNT+2    ;ADD 1 TO THE PASS COUNT
2530 014074 005537 001004          ADC      PCNT        ;MAKE IT DOUBLE PREC.
2531 014100 013700 000042    4$:     MOV      @#42,R0 ;GET MONITOR ADDRESS
2532 014104 001405          BEQ      $END1       ;IF NONE
2533 014106 000005          RESET
2534 014110 004710          $ENDAD: JSR      7,(0) ;GO TO MONITOR
2535 014112 000240 000240 000240 JSR      240,240,240 ;SAVE ROOM FOR ACT11
2536 014120 000137 003232          $END1:  JMP      ADTST    ;RETURN
2537
2538 014124 000000          .TBIT:  0           ;T BIT FLAG
2539
2540 014126 012702 000001          .LOGW:  MOV      #1,R2 ;LOG WRITE ERR
2541 014132 005003          CLIND:  CLR      R3    ;CLEAR INDEX FOR TABLE
2542 014134 000413          BR      ADDR
2543
2544 014136 012702 000400          .LOGR:  MOV      #400,R2 ;LOG WRITE ERR
2545 014142 000773          BR      CLIND
2546
2547 014144 012702 000001          .LOGWC: MOV      #1,R2  ;LOG WRITC CK ERR
2548 014150 012703 000002          SETIND: MOV      #2,R3  ;SET INDEX FOR NEXT WD
2549 014154 000403          BR      ADDR
2550
2551 014156 012702 000400          .LOGC:  MOV      #400,R2 ;LOG COMPARE ERR
2552 014162 000772          BR      SETIND
2553
2554 014164 005737 001200          ADDR:  TST      LOPCNT ;1ST TIME EROR?
2555 014170 001014          BNE     1$         ;NO DO NOT COUNT IT
2556 014172 013704 001160          MOV      UNNUM,R4   ;GET UNIT #
2557 014176 001304          ASL     R4         ;GET
2558 014200 006304          ASL     R4         ;POSITION IN
2559 014202 060304          ADD     R3,R4      ;ERR TABLE
2560 014204 060264 017072    ADD     R2,ERTAB(R4) ;TO ADD ERROR
2561 014210 004737 014224    JSR     PC,PRNT    ;TYPEOUT?
2562 014214 001402          BEQ     1$         ;YES
2563 014216 004737 015012    JSR     PC,DRP     ;SHOULD I DROP DRIVE?
2564 014222 000002          1$:     RTI
2565
2566 014224 032777 020000 164574 PRNT:  BIT      #BIT13,$SWR ;INHIBIT TYPEOUT?
2567 014232 000207          RTS      PC
    
```

K05

2568	014234	052737	000004	001122	RSREG:	BIS	#BIT2,FLAG2	:SET ERROR FLAG
2569	014242	005737	016060			TST	.HLTCT	:SHOULD WE TYPE GOOD AND BAD
2570	014246	001017				BNE	8\$:NO
2571	014250	104402	014254			TYPE	.+2	:.ASCIZ "BAD="
2572	014262	010046				MOV	BAD,-(6)	:PUT BAD ON STACK
2573	014264	104404				TYPE0		:TYPE STACK IN OCTAL
2574	014266	104402	014272			TYPE	.+2	:.ASCIZ "GOOD="
2575	014302	010146				MOV	GOOD,-(6)	:PUT GOOD ON STACK
2576	014304	104404				TYPE0		:TYPE STACK IN OCTAL
2577	014306				8\$:	TYPE	.+2	:.ASCIZ "CS1="
2578	014306	104402	014312			MOV	ARSCS1,-(6)	:PUT ARSCS1 ON STACK
2579	014320	017746	164506			TYPE0		:TYPE STACK IN OCTAL
2580	014324	104404				TYPE	.+2	:.ASCIZ "ER="
2581	014326				1\$:	MOV	ARSER,-(6)	:PUT ARSER ON STACK
2582	014326	104402	014332			TYPE0		:TYPE STACK IN OCTAL
2583	014340	017746	164502			TYPE	.+2	:.ASCIZ "CS2="
2584	014344	104404				MOV	ARSCS2,-(6)	:PUT ARSCS2 ON STACK
2585	014346				2\$:	TYPE0		:TYPE STACK IN OCTAL
2586	014346	104402	014352			MOV	ARSCS2,-(6)	:PUT ARSCS2 ON STACK
2587	014360	017746	164450			TYPE0		:TYPE STACK IN OCTAL
2588	014364	104404				BIT	#200,.HLTCT	:PRINT SECOND SET ?
2589	014366	032737	000200	016060		BNE	SEEC	:YES
2590	014374	001112				BIT	#AS,.HLTCT	:PRINT ER ?
2591	014376	032737	000100	016060		BEQ	3\$:NO
2592	014404	001410				TYPE	.+2	:.ASCIZ "AS="
2593	014406	104402	014412			MOV	ARSAS,-(6)	:PUT ARSAS ON STACK
2594	014420	017746	164424			TYPE0		:TYPE STACK IN OCTAL
2595	014424	104404				BIT	#BA,.HLTCT	:PRINT BUS ADDRESS
2596	014426	032737	000020	016060	3\$:	BEQ	4\$:NO
2597	014434	001410				TYPE	.+2	:.ASCIZ "BA="
2598	014436	104402	014442			MOV	ARSBA,-(6)	:PUT ARSBA ON STACK
2599	014450	017746	164364			TYPE0		:TYPE STACK IN OCTAL
2600	014454	104404				BIT	#DA,.HLTCT	:PRINT DA ?
2601	014456	032737	000004	016060	4\$:	BEQ	5\$:NO
2602	014464	001410				TYPE	.+2	:.ASCIZ "DA="
2603	014466	104402	014472			MOV	ARSDA,-(6)	:PUT ARSDA ON STACK
2604	014500	017746	164336			TYPE0		:TYPE STACK IN OCTAL
2605	014504	104404				BIT	#WC,.HLTCT	:PRINT WC?
2606	014506	032737	000010	016060	5\$:	BEQ	6\$:NO
2607	014514	001410				TYPE	.+2	:.ASCIZ "WC="
2608	014516	104402	014522			MOV	ARSWC,-(6)	:PUT ARSWC ON STACK
2609	014530	017746	164302			TYPE0		:TYPE STACK IN OCTAL
2610	014534	104404				BIT	#DS,.HLTCT	:DRIVE STATUS
2611	014536	032737	000040	016060	6\$:	BEQ	9\$:NO
2612	014544	001410				TYPE	.+2	:.ASCIZ "DS="
2613	014546	104402	014552			MOV	ARSDS,-(6)	:PUT ARSDS ON STACK
2614	014560	017746	164260			TYPE0		:TYPE STACK IN OCTAL
2615	014564	104404				BIT	#DB,.HLTCT	:PRINT DATA BUFFER
2616	014566	032737	000002	016060	9\$:			

2617	014574	001461				BEG	PTDONE		:NO
2618	014576	104402	014602			TYPE	.+2		:.ASCIZ " DB="
2619	014610	017746	164240			MOV	DRSDB,-(6)		:PUT DRSDB ON STACK
2620	014614	104404				TYPE0			:TYPE STACK IN OCTAL
2621	014616	000137	014740			JMP	PTDONE		:GET OUT
2622	014622	042737	000200	016060	SEEC:	BIC	#200,.HLTCT		:CLEAR COMMON BIT
2623	014630	032737	000240	016060		BIT	#DT,.HLTCT		:PRINT DRIVE TYPE?
2624	014636	001410				BEG	10\$:NO
2625	014640	104402	014644			TYPE	.+2		:.ASCIZ " DT="
2626	014652	017746	164202			MOV	DRSDT,-(6)		:PUT DRSDT ON STACK
2627	014656	104404				TYPE0			:TYPE STACK IN OCTAL
2628	014660	032737	000220	016060	10\$:	BIT	#MR,.HLTCT		:PRINT MN?
2629	014666	001410				BEG	11\$:NO
2630	014670	104402	014674			TYPE	.+2		:.ASCIZ " MR="
2631	014702	017746	164150			MOV	DRSMR,-(6)		:PUT DRSMR ON STACK
2632	014706	104404				TYPE0			:TYPE STACK IN OCTAL
2633	014710	032737	000204	016060	11\$:	BIT	#LA,.HLTCT		:PRINT LA?
2634	014716	001410				BEG	PTDONE		:NO
2635	014720	104402	014724			TYPE	.+2		:.ASCIZ " LA="
2636	014732	017746	164114			MOV	DRSLA,-(6)		:PUT DRSLA ON STACK
2637	014736	104404				TYPE0			:TYPE STACK IN OCTAL
2638	014740	032737	010000	001122	PTDONE:	BIT	#BIT12,FLAG2		:POWER FAIL TEST?
2639	014746	001111				BNE	RETT		:YES
2640	014750	104402	014754			TYPE	.+2		:.ASCIZ <15><12>"PASS "
2641	014764	013746	001006			MOV	PCNT+2,-(6)		:PUT PCNT+2 ON STACK
2642	014770	104406				TYPES			:TYPE STACK IN OCTAL - SUPRESS
2643	014772	032777	001000	164026		BIT	#BIT9,DSWR		:LOOPING ON ERROR?
2644	015000	001404				BEG	DRP		:NO
2645	015002	104402	015006			TYPE	.+2		:.ASCIZ <15><12>
2646	015012	032777	000001	164006	DRP:	BIT	#BIT0,DSWR		:DROP DRIVE?
2647	015020	001464				BEG	RETT		:NO
2648	015022	013704	001160			MOV	UNNUM,R4		:GET UNIT #
2649	015026	014737	014032			JSR	PC,GETERR		:GET ERRORS
2650	015032	063737	001202	001206		ADD	WRITER,READER		:ADD THE ERRORS
2651	015040	063737	001206	001204		ADD	READER,WCERR		
2652	015046	063737	001204	001210		ADD	WCERR,COMERR		
2653	015054	022737	000023	001210		CMP	#23,COMERR		:DROP DRIVE?
2654	015062	103043				BHIS	RETT		:NO
2655	015064	053737	001164	001124		BIS	UNCMP,DROP		:DROP DRIVE
2656	015072	104402	015076			TYPE	.+2		:.ASCIZ <15><12>"DROPPED UNIT "
2657	015116	013746	001160			MOV	UNNUM,-(6)		:PUT UNNUM ON STACK
2658	015122	104406				TYPES			:TYPE STACK IN OCTAL - SUPRESS
2659	015124	104402	000636			TYPE	.CRLF		
2660	015130	113703	001124			MOVB	DROP,R3		:GET DROPPED UNITS
2661	015134	113704	001162			MOVB	UNITSV,R4		:GET ALL DRIVES
2662	015140	020304				CMP	R3,R4		:ALL DRIVES DROPPED?
2663	015142	001003				BNE	2\$:NO
2664	015144	000000				HALT			:NO MORE DRIVES
2665	015146	000137	001234			JMP	DRBEGIN		:RESTART TEST
2666	015152	032737	100000	001126	2\$:	BIT	#BIT15,FLAG		:DID OPERATOR SELECT PATTERN
2667	015160	001002				BNE	3\$:YES
2668	015162	005037	001136			CLR	PATNU		:NO CLEAR IT
2669	015166	000137	006072		3\$:	JMP	DRXTPPR		:GET NEXT DRIVE
2670	015172	000207			RETT:	RTS	PC		

MOS

CZRSCG RH11-R503-R503/LA-R504 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 65

SEQ 0064

\$DONE - BELL AND SCOPE ROUTINE

```

2671      ;ROUTINE TO RESTORE LOADER
2672 015174 013705 015220  RLDR:  MOV      LDRI,R5      ;GET FIRST ADDRESS OF WHERE LOADER IS
2673      ;TO BE RESTORED
2674 015200 012704 017446      MOV      #17446,R4      ;ADDRESS WHERE LOADER IS STORED
2675 015204 012702 000155      MOV      #155,R2      ;WORD COUNT
2676 015210 012425      1$:  MOV      (R4)+,(R5)+      ;RESTORE
2677 015212 005302      DEC      R2
2678 015214 001375      BNE     1$
2679 015216 000000      HALT
2680 015220 017446      LDRI:  .WORD 17446      ;DONE
2681      ;FIRST ADDRESS WHERE LOADERS ARE SAVED
2682      172100
2683      000114
2684 015222 012737 015314 000114  .MAMK:  MOV      #.PARSRV,2#PARVEC
2685 015230 012737 000340 000116      MOV      #340,2#PARVEC+2 ;SET PRI LEVEL TO 7
2686 015236 013746 000004      MOV      2#4,-(SP)      ;SAVE CURRENT ERROR VECTC.
2687 015242 013746 000006      MOV      2#6,-(SP)      ;SAVE PRIORITY LEVEL
2688 015246 012737 000006 000004      MOV      #6,2#4
2689 015254 012737 000002 000006      MOV      #RTI,2#6
2690 015262 012700 172100      MOV      #PARCSR,RO      ;GET FIRST CSR ADDR
2691 015266 012702 000001      MOV      #1,R2
2692 015272 012720 000001      1$:  MOV      #1,(RO)+      ;SET ACTION ENABLE IF AVAILABLE
2693 015276 006302      ASL     R2      ;SHIFT AVAILABILITY INDICATOR
2694 015300 103374      BCC     1$
2695 015302 012637 000006      MOV      (SP)+,2#6      ;RESTORE ERROR VECTOR PRIORITY
2696 015306 012637 000004      MOV      (SP)+,2#4      ;AND INTERRUPT VECTOR
2697 015312 000207      RTS     PC
2698      ;PARITY MEMORY TRAP
2699
2700 015314 104402 000736      .PARSRV:  TYPE     ,PAREER
2701 015320 052737 004000 001122      BIS     #BIT11,FLAG2      ;SET ERROR FLAG
2702 015326 032737 000010 001122      BIT     #BIT3,FLAG2      ;WERE WE COMPARING DURING ERROR?
2703 015334 001422      BEQ     13$      ;NO
2704 015336 104402 015342      TYPE     ,+2      ;ASCIZ " ADDR="
2705 015352 005737 001212      TST     MMAVA      ;IS MEM MGMT ON?
2706 015356 001406      BEQ     12$      ;NO
2707 015360 013746 177776      MOV     PS,-(6)      ;GET PS
2708 015364 013746 001150      MOV     SAVE,-(6)      ;GET VIRTUAL ADDR
2709 015370 104412      TYPEA   ;CONVERT TO PHY AND TYPE
2710 015372 000403      BR      13$      ;CONT
2711 015374 013746 001150      12$:  MOV     SAVE,-(6)      ;GET ADDR
2712 015400 104406      TYPES   ;TYPE IT
2713 015402 032777 100000 163416 13$:  BIT     #BIT15,2SWR      ;HALT ON ERROR?
2714 015410 001401      BEQ     1$      ;NO
2715 015412 000000      HALT    ;YES
2716 015414 012706 000500      1$:  MOV     #500,SP      ;RESET STACK
2717 015420 000137 003246      JMP     EXMFLG      ;RESTART TEST
    
```

N05

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 66
 STYPE - TTY TYPEOUT ROUTINE

SEQ 0065

```

2718          .SBTTL          STYPE - TTY TYPEOUT ROUTINE
2719
2720          ; THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
2721          ; CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ADR" - TYPES THE
2722          ; MESSAGE STARTING IN LOCATION "ADR:" 2) "TYPE CHAR" - TYPES
2723          ; THE ASCII "CHAR", AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES
2724          ; THE MESSAGE WHICH IS INLINE ASCII. THE FILLER CHARACTER WHICH IS
2725          ; TYPED AFTER A LINE FEED IS IN FILCHR AND THE NUMBER OF FILLERS
2726          ; IS IN FILCHR+1.
2727
2728 015424 010446          .TYPE:  MOV    R4,-(6)          ;SAVE R4
2729 015426 010546          MOV    R5,-(6)          ;SAVE R5
2730 015430 017605 000004  MOV    @4(6),R5      ;GET ADDRESS TO BE TYPED
2731 015434 032705 177400  BIT    #177400,R5    ;IS IT A TYPED?
2732 015440 001002          BNE    1$           ;NO
2733 015442 016605 000004  MOV    4(6),R5      ;GET ADDRESS OF CHARACTER
2734 015446 105715          1$:   TSTB   (R5)       ;TERMINATOR?
2735 015450 001423          BEQ    2$           ;GET OUT IF SO
2736 015452 122715 000012  CMPB   #12,(R5)     ;IS THE CHAR A LINE FEED
2737 015456 001012          BNE    4$           ;NO - GET OUT
2738 015460 113704 001015  MOVB   FILCHR+1,R4  ;GET THE FILL COUNT
2739 015464 113777 001014 163332 5$:   MOVB   FILCHR,@TPB  ;TYPE A FILLER
2740 015472 105777 163320  TSTB   @TPS         ;DONE YET?
2741 015476 100375          BPL    -4           ;NO - WAIT
2742 015500 005304          DEC    R4           ;DEC COUNT
2743 015502 001370          BNE    5$           ;LOOP UNTIL 0
2744 015504 112577 163314 4$:   MOVB   (R5)+,@TPB  ;LOAD AND TYPE THE CHARACTER
2745 015510 105777 163302  TSTB   @TPS         ;IS THE PRINTER READY
2746 015514 100375          BPL    -4           ;WAIT UNTIL IT IS
2747 015516 000753          BR     1$           ;GET THE NEXT CHARACTER
2748 015520 017646 000004  MOV    @4(6),-(6)   ;GET ADDRESS TO BE TYPED
2749 015524 062766 000002 000006 2$:   ADD    #2,6(6)      ;ADD 2 TO THE ADDRESS
2750 015532 022666 000004  CMP    (6)+,4(6)    ;IS IT .+2?
2751 015536 001006          BNE    3$           ;NO
2752 015540 062705 000002  ADD    #2,R5        ;ADD 2 TO THE ADDRESS
2753 015544 042705 000001  BIC    #1,R5        ;BACK UP TO AN EVEN BYTE
2754 015550 010566 000004  MOV    R5,4(6)      ;RESTORE ADDRESS
2755 015554 012605 3$:   MOV    (6)+,R5      ;RESTORE R5
2756 015556 012604          MOV    (6)+,R4      ;RESTORE R4
2757 015560 000002          RTI                 ;RETURN
  
```

```

2758 .SBTTL $SCOPE - SCOPE LOOP HANDLER
2759
2760 ; THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
2761 ; LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
2762 ; "SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
2763 ; RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"
2764
2765 .SCOPE: KBDIN ; GO CHECK FOR IG
2766 015562 104442 BIT #SW14,JSWR ; LOOP ON TEST?
2767 015564 032777 040000 163234 .KIT ; LOOP ON TEST IS SET
2768 015572 001045 BNE .KIT ; LOOP ON TEST IS SET
2769 015574 000416 BR 3$ ; SKIP - NOP FOR XOR TESTER
2770 015576 013746 000004 MOV @#4,-(6) ; PUSH @#4 ON STACK
2771 015602 012737 015622 000004 MOV #4$,@#4 ; SET FOR TIMEOUT
2772 015610 005737 177060 TST @#177060 ; ERROR ON XOR?
2773 015614 012637 000004 MOV (6)+,@#4 ; POP STACK INTO @#4
2774 015620 000422 BR .SVLAD ; NO ERROR - GO TO NEXT TEST
2775 015622 022626 4$: CMP (6)+,(6)+ ; CLEAR STACK
2776 015624 012637 000004 MOV (6)+,@#4 ; POP STACK INTO @#4
2777 015630 000426 BR .KIT ; ERROR - LOOP ON TEST
2778 015632 032777 004000 163166 3$: BIT #SW11,JSWR ; KILL ITERATIONS
2779 015640 001012 BNE .SVLAD ; YES - KILL ITERATIONS
2780 015642 105737 001001 TSTB ICNT+1 ; FIRST ONE?
2781 015646 001404 BEQ 2$ ; BRANCH IF FIRST
2782 015650 123737 015734 001001 CMPB TIMES,ICNT+1 ; DONE?
2783 015656 003013 BGT .KIT ; BRANCH IF NOT
2784 015660 112737 000001 001001 2$: MOVB #1,ICNT+1 ; FIRST ITERATION
2785 015666 105237 001000 .SVLAD: INCB ICNT ; COUNT TEST NUMBERS
2786 015672 011637 001010 MOV (6),LAD ; SAVE LOOP ADDRESS
2787 015676 013777 001000 163124 MOV ICNT,@DISPLAY ; DISPLAY TEST NO. AND ITERATION COUNT
2788 015704 000002 RTI ; RETURN
2789 015706 105237 001001 .KIT: INCB ICNT+1 ; INC THE ITERATION COUNT
2790 015712 013777 001000 163110 .OVER: MOV ICNT,@DISPLAY ; SET UP DISPLAY
2791 015720 005737 001010 TST LAD ; FIRST ONE?
2792 015724 001760 BEQ .SVLAD ; YES
2793 015726 013716 001010 MOV LAD,(6) ; FUDGE RETURN ADDRESS
2794 015732 000002 RTI ; FIXES PS
2795
2796 015734 000001 TIMES: 1 ; RUN 1 TIMES

```

.SBTTL \$HLT - HLT ROUTINE (ERROR TYPEOUT)

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

2797										
2798										
2799										
2800										
2801										
2802										
2803										
2804										
2805	015736	005237	001002							
2806	015742	104442								
2807	015744	032777	020000	163054						
2808	015752	001025								
2809	015754	104402	015760							
2810	015764	011637	001012							
2811	015770	162737	000002	001012						
2812	015776	117737	163010	016060						
2813	016004	013746	0C1012							
2814	016010	104404								
2815	016012	104402	016016							
2816	016022	004737	C14234							
2817	016026	005777	162774							
2818	016032	100001								
2819	016034	000000								
2820	016036	032777	001000	162762						
2821	016044	001003								
2822	016046	105037	001001							
2823	016052	000002								
2824	016054	000137	015706							
2825										
2826	016060	000000								

```

.HLT:  INC      ERRORS      ;INC THE ERROR COUNT
      KBDIN     ;GO CHECK FOR IG
      BIT      #SW13,DSWR    ;SKIP TYPEOUT IF SET
      BNE     2$          ;SKIP TYPEOUTS
      TYPE     .+2        ;ASCIZ <15><12>
      MOV     (6),HLTADR    ;PUT ADDRESS OF INSTRUCTION ON STACK
      SUB     #2,HLTADR    ;FUDGE ADDRESS
      MOV     @HLTADR,.HLTCT ;GET HLT ARGUMENT
      MOV     HLTADR,-(6)  ;PUT HLTADR ON STACK
      TYPE    TYPE0       ;TYPE STACK IN OCTAL
      JSR     PC,RSREG     ;ASCIZ ""
      TST    DSWR        ;GO TO USER ERROR ROUTINE
      BPL    .+4         ;HALT ON ERROR
      HALT   ;SKIP IF CONTINUE
      BIT    #SW9,DSWR    ;HALT ON ERROR!
      BNE     3$          ;CHECK FOR INHIBIT LOOP ON ERROR
      CLRB   ICNT+1      ;SKIP IF LOOP ON ERROR
      RTI    ;CLEAR ITERATION COUNT
      JMP    .KIT        ;RETURN
      ;LOOP ON TEST UNTIL NO ERRORS
.HLTCT: 0 ;HLT ARGUMENT

```

```

2827          .SBTTL          SOCTAL - OCTAL TYPEOUT ROUTINE
2828
2829          : THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
2830          : ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, OR TYPE THE
2831          : 16 BITS. IT IS CALLED VIA THE TYOCT, TYPBIT, OR TYPOCS MACRO'S.
2832
2833 016062 012737 170101 016250 .TYPEB: MOV      #170101,.PR      ;SET BIT FLAG AND 16. CHARACTER COUNT
2834 016070 000411              BR          .PTIT        ;NOW TYPE IT IN BIT FORM
2835 016072 112737 000001 016250 .TYPEO: MOVB   #1,.PR        ;SET ZERO FILL SWITCH
2836 016100 000402              BR          .+6          ;SKIP
2837 016102 005037 016250          .TYPES: CLR     .PR          ;SUPRESS LEADING ZERO'S
2838 016106 112737 177772 016251          MOVB   #-6,.PR+1    ;SET COUNT
2839 016114
2840 016114 010446              MOV     R4,-(6)      ;PUSH R4 ON STACK
2841 016116 010546              MOV     R5,-(6)      ;PUSH R5 ON STACK
2842 016120 016605 000010          MOV     10(6),R5    ;GET THE DATA
2843 016124 012704 016252          MOV     #.PR+2,R4   ;SET POINTER TO FIRST ASCII CHAR.
2844 016130 105014              CLRB   (4)          ;CLEAR FIRST BYTE
2845 016132 000411              BR      .PRF        ;ROTATE FIRST BIT
2846 016134 105014          .PRL:  CLRB   (4)          ;CLEAR BYTE OF CHARACTER
2847 016136 032737 000100 016250          BIT    #100,.PR    ;BIT TYPING MODE?
2848 016144 001004              BNE    .PRF        ;YES - SKIP 2 ROTATES
2849 016146 006105              ROL    R5          ;ROTATE BIT INTO C
2850 016150 106114              ROLB   (4)         ;PACK IT
2851 016152 006105              ROL    R5          ;ROTATE BIT INTO C
2852 016154 106114              ROLB   (4)         ;PACK IT
2853 016156 006105          .PRF:  ROL    R5          ;ROTATE BIT INTO C
2854 016160 106114              ROLB   (4)         ;PACK IT
2855 016162 105714              TSTB   (4)         ;IS IT ZERO?
2856 016164 001402              BEQ    .+6         ;SKIP INC
2857 016166 105237 016250          INCB   .PR        ;SET FILL SWITCH
2858 016172 105737 016250          TSTB   .PR        ;CHECK FILL SWITCH
2859 016176 001402              BEQ    .+6         ;SKIP BITSET
2860 016200 152724 000060          BISB   #'0,(4)+   ;MAKE INTO ASCII CHAR
2861 016204 105237 016251          INCB   .PR+1      ;INC COUNT
2862 016210 001351          .PRL:  REPEAT
2863 016212 022704 016252          CMP    #.PR+2,R4  ;EMPTY BUFFER?
2864 016216 001002              BNE    .+6         ;SKIP IF NOT
2865 016220 112724 000060          MOVB   #'0,(4)+   ;LOAD 1 ZERO
2866 016224 105014              CLRB   (4)        ;NULL TERMINATOR
2867 016226 104402 016252          TYPE   (.PR+2)    ;TYPE IT
2868 016232 012605              MOV    (6)+,R5    ;POP STACK INTO R5
2869 016234 012604              MOV    (6)+,R4    ;POP STACK INTO R4
2870 016236 016666 000002 000004          MOV    2(6),4(6)  ;GET RID OF
2871 016244 012616              MOV    (6)+,(6)   ;DATA WORD
2872 016246 000002              RTI
2873
2874 016250 000012          .PR:   .BLKW   12      ;COUNT, SWITCH, AND OUTPUT BUFFER

```

E06

CZRSCG RH11-R503-R503 LA-R504 DATA AND RELIABILITY TEST
CZRSCG.F11 24-JAN-78 07:42MACY11 30A(1052) 24-JAN-78 07:42 PAGE 70
\$POWER - POWER DOWN AND UP ROUTINES

SEQ 0069

```

2875          .SBTTL          $POWER - POWER DOWN AND UP ROUTINES
2876
2877          ; THIS IS THE POWER FAIL ROUTINE WHICH WILL SAVE ALL
2878          ; THE GENERAL REGISTERS AND USER DEFINED REGISTERS THEN
2879          ; WAIT FOR POWER TO GO DOWN AND BE RESTORED.
2880          ; IF THERE ISN'T ENOUGH TIME FOR SAVING ALL THE REGISTERS,
2881          ; THE PROGRAM WILL HALT AT '.ILLUP'.
2882
2883          016274 012777 016422 000126 .POWER: MOV      #.ILLUP,@.PUVEC ;SET FOR FAST UP
2884          016302 012777 000340 000122          MOV      #340,@.PUVECS+2 ;PRIO:7
2885          016310 010046          MOV      R0,-(6) ;PUSH R0 ON STACK
2886          016312 010146          MOV      R1,-(6) ;PUSH R1 ON STACK
2887          016314 010246          MOV      R2,-(6) ;PUSH R2 ON STACK
2888          016316 010346          MOV      R3,-(6) ;PUSH R3 ON STACK
2889          016320 010446          MOV      R4,-(6) ;PUSH R4 ON STACK
2890          016322 010546          MOV      R5,-(6) ;PUSH R5 ON STACK
2891          016324 010637 016426          MOV      SP,.SAVR6 ;SAVE SP
2892          016330 012777 016340 000072          MOV      #.POWUP,@.PUVEC ;SET UP VECTOR
2893          016336 000000          HALT ;WAIT FOR PF
2894
2895          016340 013706 016426          .POWUP: MOV      .SAVR6,SP ;GET SP
2896          016344 005001          CLR      R1 ;WAIT LOOP FOR THE TTY
2897          016346 005201          IS: INC      R1 ;WAIT FOR THE INC
2898          016350 001376          BNE     IS ;OF WORD
2899          016352 012605          MOV      (6)+,R5 ;POP STACK INTO R5
2900          016354 012604          MOV      (6)+,R4 ;POP STACK INTO R4
2901          016356 012603          MOV      (6)+,R3 ;POP STACK INTO R3
2902          016360 012602          MOV      (6)+,R2 ;POP STACK INTO R2
2903          016362 012601          MOV      (6)+,R1 ;POP STACK INTO R1
2904          016364 012600          MOV      (6)+,R0 ;POP STACK INTO R0
2905          016366 012737 016274 000024          MOV      #.POWER,@#24 ;SET UP THE POWER DOWN VECTOR
2906          016374 012737 000340 000026          MOV      #340,@#26 ;PRIO:7
2907          016402 104402 016406          TYPE    +2 ;ASCIZ <15><12>"POWER"
2908          016416 000137 007016          JMP     ↑IMUP ;JMP TO USER ADDRESS
2909
2910          016422 000000          .ILLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
2911          016424 000776          BR      -2 ;BEFORE THE POWER DOWN WAS COMPLETE
2912
2913          016426 000000          .SAVR6: 0 ;PUT THE SP HERE
2914          016430 000024 000026          .PUVEC: 24,26 ;POWER UP VECTOR

```

F06

2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970

016434
016434
016436
016440
016444
016450
016456
016464
016472
016474
016476
016500
016502
016504
016506
016512
016516
016520
016524
016526
016530
016532
016534
016536
016542
016544
016546
016550
016552
016554
016556
016560
016562
016564
016566
016570
016572
016574
016576
016600
016604
016612
016616
016624
016626
016632
016634
016636

.SBTTL

STYPER - 18 BIT ADDRESS TYPER

: THIS ROUTINE TAKES 2 ARGUMENTS OFF THE STACK (OLD
: SP AND ADDRESS) AND, USING THE MEMORY MANAGEMENT REGISTERS, TYPES
: THE ADDRESS SUPPLIED IN 18 BIT FORM. THIS ROUTINE IS LINKED
: VIA THE 'TYPADR' MACRO.

.TYPEA:

MOV R4,-(6)
MOV R5,-(6)
MOV 12(6),R5
MOV 10(6),R4
MOV 6(6),10(6)
MOV 4(6),6(6)
MOV 2(6),4(6)
MOV (6)+,(6)
MOV R3,-(6)
SWAB R5
ROR R5
ROR R5
ROR R5
BIC #177771,R5
MOV .SATAB(5),R5
MOV R4,R3
BIC #160000,R4
SWAB R3
RCR R3
ROR R3
ROR R3
ROR R3
BIC #177761,R3
ADD R3,R5
MOV (5),R5
ASL R5
ASL R5
ASL R5
ASL R5
CLR R3
ASL R5
ROL R3
ASL R5
ROL R3
ADD R4,R5
ADC R3
ASL R5
ROL R3
MOVB R3,.PR+2
ADD #0,.PR+2
MOV #.PR+3,R4
MOV #175401,.PR
MOV (6)+,R3
JMP .PRL

: PUSH R4 ON STACK
: PUSH R5 ON STACK
: R5 - OLD PS WITH PREVIOUS MODE
: R4 - ADDRESS TO BE DECODED AND TYPED
: MOVE
: DOWN
: FOUR
: WORDS
: PUSH R3 ON STACK
: GET THE
: 2 PREVIOUS
: MODE BITS
: INTO POSITION
: TO USE AS AN OFFSET
: R5 - SPACE ADDRESS FOR MM
: R3 - REGISTER OFFSET
: CLEAR THE MM REG SELECT BITS
: NOW MAKE
: MM REG
: SELECT BITS
: INTO AN
: OFFSET
: CLEAR THE JUNK BITS
: ADD THE OFFSET TO THE TABLE
: GET THE ISAR DATA
: THIS IS
: TO SHIFT
: THE SEGMENT
: ADDRESS
: INTO AN
: AN 18 BIT
: ADDRESS
: POSITION
: WITH R3 CONTAINING
: THE UPPER 2 BITS
: AND R5 CONTAINING
: THE 16 BIT ADDRESS
: THEN SHIFT FOR TYPING
: GET THE FIRST NUMBER FROM R3
: MAKE IT INTO A NUMBER
: FUDGE IN THE POINTER
: AND THE FLAGS (FILL & 5 BYTES)
: POP STACK INTO R3
: DECODE AND TYPE THE REST

.SATAB: 172340
172240
172340

: KISARO
: SISARO
: KISARO - NEVER USED

G06

CZRSCG RH11-R503-R503 LA-R504 DATA AND RELIABILITY TEST
CZRSCG.F11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 72
\$TYPEA - 18 BIT ADDRESS TYPED

SEQ 0071

2971 016640 177640

177640

:UISARD

H06

.SBTTL \$TRAP - TRAP HANDLER

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

2972			
2973			
2974			
2975			
2976			
2977			
2978			
2979	016642	011646	
2980	016644	162716	000002
2981	016650	017616	000000
2982	016654	062716	112262
2983	016660	013607	
2984			
2985	016662	015562	
2986	016664	015424	
2987	016666	016072	
2988	016670	016102	
2989	016672	021364	
2990	016674	016434	
2991	016676	006124	
2992	016700	006146	
2993	016702	017304	
2994	016704	016730	
2995	016706	021216	
2996	016710	000316	
2997	016712	014126	
2998	016714	014136	
2999	016716	014144	
3000	016720	014156	
3001	016722	017214	
3002	016724	017132	
3003	016726	020000	

```

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

```

.SCOPE	=	TRAP+0	(104400)
.TYPE	=	TRAP+2	(104402)
.TYPE0	=	TRAP+4	(104404)
.TYPES	=	TRAP+6	(104406)
.TYPED	=	TRAP+10	(104410)
.TYPEA	=	TRAP+12	(104412)
.ERCLR	=	TRAP+14	(104414)
.DKCMD	=	TRAP+16	(104416)
.RDOCT	=	TRAP+20	(104420)
.RDLIN	=	TRAP+22	(104422)
.UPDAT	=	TRAP+24	(104424)
.CLRDV	=	TRAP+26	(104426)
.LOGW	=	TRAP+30	(104430)
.LOGR	=	TRAP+32	(104432)
.LOGWC	=	TRAP+34	(104434)
.LOGC	=	TRAP+36	(104436)
.CNTLU	=	TRAP+40	(104440)
.KBDIN	=	TRAP+42	(104442)
.SUSWR	=	TRAP+44	(104444)

```

3004          .SBTTL          SRDLIN - TTY INPUT ROUTINE
3005
3006          :THIS ROUTINE INPUTS A LINE TERMINATED BY A RETURN INTO ADDRESS
3007          :INPUT AND RETURNS A LINE FEED. THE BUFFER HAS A NULL TERMINATOR
3008          :INSTEAD OF THE RETURN. RUBOUTS ARE HANDLED BY RETYPING
3009          :THE LINE. BUFFER OVERFLOW ERRORS LIKE A RUBOUT.
3010
3011          .RDLIN: MOV      R5, -(6)          ;SAVE R5
3012          1$:  MOV      #INPUT, R5          ;GET ADDRESS
3013          2$:  CMP      #INPUT+16., R5      ;BUFFER FULL?
3014          BEQ      4$                      ;YES - TYPE "?"
3015          TSTB     @#177560                ;WAIT FOR
3016          BPL      -4                      ;A CHARACTER
3017          MOVB     @#177562, (5)           ;GET CHARACTER
3018          BICB     #200, (5)              ;GET RID OF JUNK
3019          CMPB     #25, (5)               ;IS IT A ?U
3020          BNE      5$                      ;BRANCH IF NOT
3021          TYPE     .+2                     ;ASCIZ "?U" <15> <12>
3022          BR       1$                      ;START OVER
3023          5$:  CMPB     #177, (5)          ;IS IT A RUBOUT
3024          BNE      3$                      ;SKIP IF NOT
3025          4$:
3026          TYPE     .+2                     ;ASCIZ "?" <15> <12>
3027          BR       1$                      ;ZAP THE BUFFER AND LOOP
3028          3$:  MOVB     (5), #0             ;SET UP FOR TYPING
3029          TYPE     3$+2                    ;ECHO IT
3030          CMPB     #15, (5)+              ;CHECK FOR RETURN
3031          BNE      2$                      ;LOOP IF NOT RETURN
3032          TYPE     .12                     ;TYPE A LINE FEED
3033          MOV      (6)+, R5                ;RESTORE R5
3034          RTI
3035
3036          INPUT: .BLKB 16.                  ;TTY INPUT AREA
3037          ERTAB: .BLKW 16.
3038
3039          .KBDIN: TST      42                ;GOT XXDP OR ACT
3040          BNE      OKT                      ;YES, GET OUT
3041          CMP      #SWREG, SWR              ;GOT SWITCH-LESS MACHINE?
3042          BNE      OKT                      ;NO GET OUT
3043          TSTB     @TKS                     ;HAVE A CHARACTER
3044          BPL      OKT                      ;NO GET OUT
3045          MOV      @TKB, .MSG               ;STRIP OFF GARBAGE
3046          BIC      #177600, .MSG           ;DO WE HAVE A ?G
3047          CMPB     #7, .MSG                 ;NO GET OUT
3048          BNE      OKT                      ;ASCIZ <15> <12> "?G"
3049          TYPE     .+2
3050          .CNTLU:
3051          TYPE     .+2                       ;ASCIZ <15> <12> "SWR= "
3052          MOV      SWREG, -(6)              ;PUT SWREG ON STACK
3053          TYPE0    ;TYPE STACK IN OCTAL
3054          TYPE     .+2                       ;ASCIZ " NEW= "
3055          RDOCT
3056          MOV      (SP)+, .MSG              ;GET NEW VALUE OFF STACK
3057          TST      CTN                      ;DID HE TYPE <CR> OF 000000?
3058          BEQ      OKT                      ;DONT CHANGE IF <CR>
3059          MOV      .MSG, SWREG              ;CHANGE VALUE OF SWREG

```

JOB

CZRSCG RH11-R503-R503/LA-R504 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 75
SRDLIN - TTY INPUT ROUTINE

SEQ 0074

```

3060 017276 000002
3061
3062 017300 000000
3063 017302 000000
3064
3065
3066
3067
3068
3069 017304 011646 000004 000002
3070 017306 016666
3071 017314 010146
3072 017316 010246
3073 017320 010346
3074 017322 104422
3075 017324 005001
3076 017326 005037 017302
3077 017332 012703 017052
3078 017336 112302
3079 017340 122702 000015
3080 017344 001421
3081 017346 122702 000060
3082 017352 003024
3083 017354 122702 000067
3084 017360 002421
3085 017362 006002
3086 017364 006002
3087 017366 006002
3088 017370 006101
3089 017372 006102
3090 017374 006101
3091 017376 006102
3092 017400 006101
3093 017402 005237 017302
3094 017406 000753
3095 017410 010166 000012
3096 017414 012603
3097 017416 012602
3098 017420 012601
3099 017422 000002
3100
3101 017424
3102 017424 104402 017430
3103 017434 000732
3104 017436 000000
3105
3106
    
```

```

OKT: RTI ;ALL DONE-EXIT

.MSG: 0
CTN: 0
.SBTTL SRDOCT - OCTAL INPUT ROUTINE

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

.RDOCT: MOV (6),-(6) ;MOVE THE PC
MOV 4(6),2(6) ;MOVE THE PS
MOV R1,-(6) ;PUSH R1 ON STACK
MOV R2,-(6) ;PUSH R2 ON STACK
MOV R3,-(6) ;PUSH R3 ON STACK
4$: RDLIN ;READ A LINE INTO INPUT
CLR R1 ;INIT DATA WORD
CLR CTN ;CLEAR COUNT WORD
MOV #INPUT,R3 ;INIT POINTER
1$: MOVB (3)+,R2 ;GET A BYTE
CMPB #15,R2 ;WAS IT A CR?
BEQ 2$ ;GET OUT IF YES
CMPB #'0,R2 ;CHECK FOR 0 OR GREATER
BGT 3$ ;ERROR - LESS THAN 0
CMPB #'7,R2 ;CHECK FOR 7 OR LESS
BLT 3$ ;ERROR - GREATER THAN 7
ROR R2 ;GET
ROR R2 ;INTO
ROR R2 ;POSITION
ROL R1 ;FIRST BIT
ROL R2 ;GET
ROL R1 ;SECOND BIT
ROL R2 ;GET
ROL R1 ;THIRD BIT
INC CTN ;YES HE TYPED SOMETHING
BR 1$ ;LOOP
2$: MOV R1,12(6) ;SAVE THE RESULT
MOV (6)+,R3 ;POP STACK INTO R3
MOV (6)+,R2 ;POP STACK INTO R2
MOV (6)+,R1 ;POP STACK INTO R1
RTI ;RETURN

3$: TYPE 4$+2 ;.ASCIZ "?"<15><12>
BR 4$ ;TRY AGAIN

OUTBUF: 0
    
```

;NOTE FOR PROGRAMMER***** PROGRAM AT THIS POINT CAN NOT EXCEED A PC OF 17444*****

K06

CZRSCG RH11-RS03-RS03-LA-RS04
CZRSCG.F11 24-JAN-78 07:42

DATA AND RELIABILITY TEST

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 76

SEQ 0075

\$RDOCT - OCTAL INPUT ROUTINE

```

3107      020000      . = 20000
3108      ;NOTE ALL THIS CODE GETS DESTROYED WHEN PATTERN IS WRITTEN
3109
3110 020000 032737 000001 020116 .SUSWR: BIT #BIT0,SWI
3111 020006 001037      BNE XXX
3112 020010 013746 000006      MOV 6,-(SP) ;SAVE 6 ON STACK
3113 020014 013746 000004      MOV 4,-(SP) ;SAVE 4 ON STACK
3114 020020 012737 020040 000004      MOV #1,4 ;SET UP TRAP ADDRESS
3115 020026 022777 177777 160772      CMP #-1,SWR ;TEST 177570
3116 020034 001402      BEQ 2$ ;FAKE OUT
3117 020036 000407      BR 3$ ;HARDWARE AVAILABLE
3118 020040 022626      1$: CMP (SP)+,(SP)+ ;ADJUST STACK
3119 020042 012737 000176 001026 2$: MOV #SWREG,SWR ;SET UP SOFTWARE REGISTERS
3120 020050 012737 000174 001030      MOV #DISPRG,DISPLAY
3121 020056 022737 000176 001026 3$: CMP #SWREG,SWR ;1ST TIME THRU?
3122 020064 001004      BNE 4$ ;NO CHANGE STILL 177570
3123 020066 005737 000042      TST 42 ;ANY XXDP OR ACT
3124 020072 001001      BNE 4$ ;SWR=000000
3125 020074 104440      CNTLU ;GET INITIAL SETTINGS
3126 020076 012637 000004      4$: MOV (SP)+,4 ;REPLACE 4 FROM STACK
3127 020102 012637 000006      MOV (SP)+,6 ;REPLACE 6 FROM STACK
3128 020106 052737 000001 020116 XXX: BIS #BIT0,SWI ;SET THE BEENHEREBIT
3129 020114 000002      RTI ;ALL DONE
3130
3131 020116 000000      SWI: 0
3132
3133
3134
3135 ;ROUTINE TO SAVE ABS LOADER
3136 020120 012700 017776      LDR: MOV #17776,R0
3137 020124 012737 020144 000004      MOV #2$,4 ;SET TIME OUT TRAP VECTOR
3138 020132 012737 000340 000006      MOV #340,6
3139 020140 005720      TST (R0)+
3140 020142 000776      BR -2
3141 020144 022626      2$: CMP (SP)+,(SP)+
3142 020146 012737 000006 000004      MOV #6,4
3143 020154 005037      CLR 6
3144 020160 162700      SUB #334,R0 ;POINT R0 BACK TO LOADER
3145 020164 010037 015220      MOV R0,LDR1 ;SAVE FOR RESTORE ROUTINE
3146 020170 012702 000155      MOV #155,R2 ;WORD COUNT
3147 020174 012703 017446      MOV #17446,R3 ;WHERE LOADER IS TO BE STORED
3148 020200 012023      1$: MOV (R0)+,(R3)+ ;STORE LOADER
3149 020202 005302      DEC R2
3150 020204 001375      BNE 1$
3151 020206 000207      RTS PC ;RETURN
3152
3153
3154 ; -A- PORT SIZE
3155
3156 020210 052737 020000 001122 SIZZAP: BIS #BIT13,FLAG2 ;SET MAPPING BIT
3157 020216 042737 000500 001126      BIC #500, FLAG ;CLEAR FLAG BEFORE READ
3158 020224 042737 000460 001122      BIC #460, FLAG2 ;CLEAR FLAG2 BEFORE READ
3159 020232 042737 003400 001172      BIC #3400, CMD ;CLEAR CMD BEFORE READ
3160 020240 004737 001702      JSR PC,DRVENO ;FIND DRIVE
3161 020244 012737 000002 001224      MOV #2,WORK1 ;START WITH ONE 4K BUFFER
3162 020252 012737 000001 001070      MOV #1,STAMEM ;FIRST 4K BANK

```

SRDOCT - OCTAL INPUT ROUTINE

3163	020260	012737	057476	001140		MOV	#57476, BUF	:GET STARTING ADDR. 5K
3164	020266	012737	000001	001130		MOV	#1, WRDCT	:LOAD WC
3165	020274	005037	001134			CLR	DMA	:LOAD DA
3166	020300	012777	000040	160526		MOV	#40, ARSCS2	:CLEAR ALL RS REG
3167	020306	013777	001160	160520		MOV	UNNUM, ARSCS2	:GET DRIVE #
3168	020314	012737	000071	001172		MOV	#71, CMD	:DO A READ
3169	020322	104416			4\$:	DKCMD		:NOW
3170	020324	105777	160502		1\$:	TSTB	ARSCS1	:DONE YET?
3171	020330	100375				BPL	1\$:NO
3172	020332	032777	004000	160474		BIT	#4000, ARSCS2	:DID NEM SET?
3173	020340	001012				BNE	SIZ1	:YES
3174	020342	005777	160464			TST	ARSCS1	:ANY ERRORS?
3175	020346	100005				BPL	3\$:NO
3176	020350	012737	000006	001106		MOV	#6, SIZEAP	:GET SIZE OF BUFFER
3177	020356	000137	020572			JMP	ARSCS1	:FOR USER IF HE WISHES IT
3178	020362	104424			3\$:	UPDAT		:GET NEXT 4K BANK
3179	020364	000756				BR	4\$:TEST BANK
3180	020366	005337	001224		SIZ1:	DEC	WORK1	:DEC SIZE OF BUFFER
3181	020372	013737	001224	001106		MOV	WORK1, SIZEAP	:LOAD SIZE OF A BUFFER
3182	020400	104402	020404			TYPE	+2	:.ASCIZ <15><12> "PORT -A- DATA BUFFER 4K TO "
3183	020446	004737	021156			JSR	PC, SIZPR	

; -B- PORT SIZE

```

3184
3185
3186 020452 012737 000001 001224 SIZZBP: MOV #1,WORK1 ;START WITH ONE 4K BUFFER
3187 020460 042737 000500 001126 BIC #500, FLAG ;CLEAR FLAG BEFORE READ
3188 020466 042737 000460 001122 BIC #460, FLAG2 ;CLEAR FLAG2 BEFORE READ
3189 020474 042737 003400 001172 BIC #3400, CMD ;CLEAR CMD BEFORE READ
3190 020502 012737 037476 001140 MOV #37476, BUF ;GET STARTING ADDR. 4K
3191 020510 012737 000001 001130 MOV #1,WRDCT ;LOAD WC
3192 020516 005037 001134 CLR DMA ;LOAD DA
3193 020522 012777 000040 160304 MOV #40,DRSCS2 ;CLEAR ALL RS REG
3194 020530 013777 001160 160276 MOV UNNUM,DRSCS2 ;GET DRIVE #
3195 020536 012737 002071 001172 MOV #2071,CMD ;DO A READ
3196 020544 104416 SIZ2: DKCMD ;NOW
3197 020546 105777 160260 IS: TSTB DRSCS1 ;DONE YET?
3198 020552 100375 BPL 1$ ;NO
3199 020554 032777 004000 160252 BIT #4000,DRSCS2 ;DID NEM SET?
3200 020562 001067 BNE SIZB1 ;YES
3201 020564 005777 160242 TST DRSCS1 ;ANY ERRORS?
3202 020570 100047 BPL SIZ3 ;NO
3203
3204 020572 104402 020576 SIZERR: TYPE ;.ASCIZ <15><12>"WILL NOT CONTINUE TO SIZE MEMORY BECAUS
3205 020662 012737 000006 001110 MOV #6,SIZEBP ;GIVE PROGRAM A BUFFER
3206 020670 104060 HLT !BA!DS ;YOU CAN ENTER CONVERSATION MODE
3207 020672 052737 000001 001122 BIS #BIT0,FLAG2 ;BEEN HERE BEFORE FLAG
3208 020700 042737 020000 001122 BIC #BIT13,FLAG2 ;CLEAR MAPPING FLAG
3209 020706 000000 HALT ;OR GO TO DZRSB
3210 020710 032737 000400 001122 SIZ3: BIT #BIT8,FLAG2 ;FOUND MEMORY YET?
3211 020716 001006 BNE SIZB3 ;YES
3212 020720 052737 000400 001122 BIS #BIT8,FLAG2 ;SET FOUND MEM FLAG
3213 020726 013737 001224 001074 MOV WORK1,STBCOM ;GET 1ST BANK

```

N06

```

3214 020734 104424          SIZB3: UPDAT          ;GET NEXT 4K BANK
3215 020736 000702          BR              ;TEST BANK
3216 020740 000404          BR              SIZ2
3217 020742 032737 000400 001122 SIZB1: BIT          #BIT8,FLAG2
3218 020750 001771          BEQ          SIZB2
3219 020752 005337 001224          DEC          WORK1
3220 020756 013737 001224 001110 SIZB2: MOV          WORK1,SIZEBP
3221 020764 032737 000400 001122 BIT          #BIT8,FLAG2
3222 020772 001017          BNE          6$
3223 020774 104402 021000          TYPE
3224 021030 000442          BR              1$
3225 021032          6$:
3226 021032 104402 021036          TYPE          .+2
3227 021066 013737 001074 001224 MOV          $TBCOM,WORK1
3228 021074 005001          CLR          R1
3229 021076 005002          CLR          R2
3230 021100 004737 021164          JSR          PC, SIZP
3231 021104 104402 021110          TYPE          .+2
3232 021116 013737 001110 001224 MOV          $SIZEBP,WORK1
3233 021124 004737 021156          JSR          PC, SIZPR
3234 021130 052737 000100 001126 1$: BIS          #BIT6,FLAG
3235 021136 042737 020000 001122 BIC          #BIT13,FLAG2
3236 021144 052737 000002 001122 BIS          #BIT1,FLAG2
3237 021152 000137 001552          JMP          CALM
3238
3239 021156 005001          SIZPR: CLR          R1
3240 021160 012702 000004          MOV          #4,R2
3241 021164 062701 000001          SIZP: ADD          #1,R1
3242 021170 062702 000004          ADD          #4,R2
3243 021174 020137 001224          CMP          R1,WORK1
3244 021200 001371          BNE          SIZP
3245 021202 010246          MOV          R2,-(6)
3246 021204 104410          TYPED
3247 021206 104402 021212          TYPE
3248 021214 000207          RTS          PC
3249
3250          ;ADD 4K TO TEST ADDR
3251
3252 021216 005237 001224          .UPDAT: INC          WORK1
3253 021222 062737 020000 001140 ADD          #2000,BUF
3254 021230 005737 001212          TST          MMAVA
3255 021234 001437          BEQ          3$
3256 021236 022737 000037 001224 CMP          #37, WORK1
3257 021244 001440          BEQ          1$
3258 021246 022737 000027 001224 CMP          #27, WORK1
3259 021254 002004          BGE          10$
3260 021256 052737 001400 001172 BIS          #1400, CMD
3261 021264 000002          RTI
3262 021266 022737 000017 001224 10$: CMP          #17, WORK1
3263 021274 002007          BGE          11$
3264 021276 052737 001000 001172 BIS          #1000, CMD
3265 021304 042737 000400 001172 BIC          #400, CMD
3266 021312 000002          RTI
3267 021314 022737 000007 001224 11$: CMP          #7, WORK1
3268 021322 002010          BGE          2$
3269 021324 052737 000400 001172 BIS          #400, CMD

```

```

;GET NEXT 4K BANK
;TEST BANK
;FOUND MEMORY?
;NO
;DEC SIZE OF BUFFER
;LOAD SIZE OF B BUFFER
;FOUND MEMORY?
;YES
;.ASCIZ <15><12>"NO MEMORY ON -B- PORT"
;.ASCIZ <15><12> "PORT -B- DATA BUFFER"
;.ASCIZ " TO "
;SET MULTI PORT FLAG
;CLEAR MAPPING FLAG
;SET BEEN HERE FLAG
;CAL BUFFER AND WC
;INIT SETUP
;SETUP FOR BANK NO
;SETUP FOR SIZE FO MEMORY
;IS THIS THE RIGHT SIZE?
;NO
;PUT R2 ON STACK
;TYPE STACK IN DECIMAL
;.ASCIZ "K"
;RETURN
;INC BANK #
;UPDATE BY 4K
;MEMORY MANAGEMENT AVAILABLE?
;NO
;BANK 37?
;YES-STOP SIZING
;BANK 30?
;NO
;YES-ENABLE A16 AND A17
;CONTINUE
;BANK 20?
;NO
;YES-ENABLE A17
;DISABLE A16
;CONTINUE
;BANK 10?
;NO
;YES-ENABLE A16

```

3270	021332	000002				RTI			: CONTINUE
3271	021334	022737	177476	001140	3\$:	CMP	#177476, BUF		: EXCEEDED MEM YET?
3272	021342	001401				BEQ	1\$: YES
3273	021344	000002			2\$:	RTI			: CONTINUE
3274	021346	062716	000002		1\$:	ADD	#2, (6)		: UPDATE RETURN PC BY 2
3275	021352	042737	003400	001172		BIC	#3400, CMD		: CLEAR CMD AFTER FINISHING
3276	021360	000002				RTI			
3277									
3278	021362	021364				.TYPED			: TYPED = TRAP+46 (104446)
3279						.SBTTL	STYPED - CONVERT		: BINARY TO DECIMAL AND TYPE ROUTINE
3280									
3281	021364	012737	100040	021612		.TYPED: MOV	#100040, .DSIGN		: SET BLANK SWITCH AND SIGN
3282	021372	010046				MOV	R0, -(6)		: PUSH R0 ON STACK
3283	021374	010146				MOV	R1, -(6)		: PUSH R1 ON STACK
3284	021376	010246				MOV	R2, -(6)		: PUSH R2 ON STACK
3285	021400	010346				MOV	R3, -(6)		: PUSH R3 ON STACK
3286	021402	010546				MOV	R5, -(6)		: PUSH R5 ON STACK
3287	021404	012737	100040	021612		MOV	#100040, .DSIGN		: SET BLANK SWITCH AND SIGN
3288	021412	016605	070016			MOV	16(6), R5		: GET DATA TO BE TYPED
3289	021416	100004				BPL	1\$: BR IF INPUT IS POS.
3290	021420	005405				NEG	R5		: MAKE THE BINARY NUMBER POS.
3291	021422	112737	000055	021612		MOV	#'-, .DSIGN		: MAKE THE ASCII NUMBER NEG.
3292	021430	005000			1\$:	CLR	R0		: ZERO THE CONSTANTS INDEX
3293	021432	012703	021602			MOV	#.DBLK, R3		: SETUP THE OUTPUT POINTER
3294	021436	112723	000040			MOV	#', (R3)+		: SET THE FIRST CHARACTER TO A BLANK
3295	021442	005002			2\$:	CLR	R2		: CLEAR THE BCD NUMBER
3296	021444	016001	021572			MOV	.DTBL(R0), R1		: GET THE CONSTANT
3297	021450	160105			3\$:	SUB	R1, R5		: FORM THIS BCD DIGIT
3298	021452	002402				BLT	4\$: BR IF DONE
3299	021454	005202				INC	R2		: INCREASE THE BCD DIGIT BY 1
3300	021456	000774				BR	3\$		
3301	021460	060105			4\$:	ADD	R1, R5		: ADD BACK THE CONSTANT
3302	021462	005702				TST	R2		: CHECK IF BCD DIGIT=0
3303	021464	001003				BNE	5\$: FALL THROUGH IF 0
3304	021466	105737	021613			TSTB	.DSIGN+1		: STILL DOING LEADING 0'S?
3305	021472	100410				BMI	7\$: BR IF YES
3306	021474	106337	021613		5\$:	ASLB	.DSIGN+1		: MSD?
3307	021500	103003				BCC	6\$: BR IF NO
3308	021502	113763	021612	177777		MOV	.DSIGN, -1(R3)		: YES--SET THE SIGN
3309	021510	052702	000060		6\$:	BIS	#'0, R2		: MAKE THE BCD DIGIT ASCII
3310	021514	052702	000040		7\$:	BIS	#', R2		: MAKE IT A SPACE IF NOT ALREADY A DIGIT
3311	021520	110223				MOV	R2, (R3)+		: PUT THIS CHARACTER IN THE OUTPUT BUFFER
3312	021522	005720				TST	(R0)+		: JUST INCREMENTING
3313	021524	020027	000010			CMP	R0, #10		: CHECK THE TABLE INDEX
3314	021530	002744				BLT	2\$: GO DO THE NEXT DIGIT
3315	021532	003002				BGT	8\$: GO TO EXIT
3316	021534	010502				MOV	R5, R2		: GET THE LSD
3317	021536	000764				BR	6\$: GO CHANGE TO ASCII
3318	021540	105013			8\$:	CLRB	(R3)		: SET THE TERMINATOR
3319	021542	012605				MOV	(6)+, R5		: POP STACK INTO R5
3320	021544	012603				MOV	(6)+, R3		: POP STACK INTO R3
3321	021546	012602				MOV	(6)+, R2		: POP STACK INTO R2
3322	021550	012601				MOV	(6)+, R1		: POP STACK INTO R1
3323	021552	012600				MOV	(6)+, R0		: POP STACK INTO R0
3324	021554	016666	000002	000004		MOV	2(6), 4(6)		: FUDGE DATA
3325	021562	012616				MOV	(6)+, (6)		: OFF STACK

C07

CZRSCG RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCG.P11 24-JAN-78 07:42

MACY11 30A(1052) 24-JAN-78 07:42 PAGE 81
\$TYPED - CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0080

3326	021564	104402	021602	TYPE	..DBLK	;NOW TYPE THE NUMBER
3327	021570	000002		RTI		;RETURN
3328						
3329	021572	023420	001750	000144	.DTBL:	10000.,1000.,100.,10.
3330	021600	000012				
3331	021602	000004			.DBLK:	.BLKW 4
3332	021612	000000			.DSIGN:	0
3333						
3334		000001			.END	

. ABS. 021614 000

ERRORS DETECTED: 0

CZRSCG.BIN,CZRSCG.LST/CRF/SOL/NL:TOC=CZRSCG.SML,CZRSCG.P11
RUN-TIME: 8 12 1 SECONDS
RUN-TIME RATIO: 74/21=3.4
CORE USED: 22K (43 PAGES)

C08

EOF1CZRSCGSEQ

00010000

780330

PDP10 411