

RH11-RS03

RH11-RS03/4 DATA RELIABILITY
CZRSCF0

AH-9319F-MC
COPYRIGHT © 73-78
FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

Microfiche grid containing multiple frames of data, including tables and charts. The content is too small to transcribe accurately.



801

EOF1CZTRABSEQ
GE 2 PDP10 411
CZRSCF.P11

00010000
CZRSCF

16-DEC-77 15:10

780223 PDP10 411 D:HDR1CZRSCFSEQ
RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST

00010000 780223
MACY11 30(1046) 20-DEC-77 15:36 PA
SEQ 0001

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

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-9318F-MC
PRODUCT NAME: CZRSCFO RH11-RS03/RS04 DATA RELIABILITY
DIAGNOSTIC
PROGRAM DATE: FEB 1978
MAINTAINER: DIAGNOSTIC GROUP
MODIFIED BY: BILL SCHLITZKUS

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

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF
SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR
ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1973, 1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL PDP UNIBUS MASSBUS
DEC DECUS DECTAPE

CONTENTS

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

1
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

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

AC-9318F-MC CZRSCF RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 3
DESCRIPTION

1. ABSTRACT

THIS DIAGNOSTIC WAS DESIGNED TO TEST RS03, RS03/LA AND RS04 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 (RS03/LA OR RS04) 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 RS03, RS03/LA OR AN RS04 DISK.

2.2 PRELIMINARY PROGRAMS

DZRSB

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

AC-9318F-MC CZRSCF RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 4
DESCRIPTION

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

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.

183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214

AC-9318F-MC CZRSCF 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.

AC-9318F-MC CZRSCF RH11-RS03/LA-RS04 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

215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247

AC-9318F-MC CZRSCF 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

AC-9318F-MC CZRSCF RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 8
DESCRIPTION

THE PROGRAM WILL NOW ASK SEVERAL QUESTIONS, THE TABLE BELOW WILL HELP YOU ANSWER THE QUESTIONS.

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

NOTE: TYPE ONLY NUMBERS SHOWN!!!

1. -A- PORT? (Y OR N)

THIS GIVES YOU THE OPTION TO TEST -A- OR -B- PORT USING THE DATA TEST.

IF THE ANSWER TO THIS QUESTION IS YES, THE FOLLOWING QUESTION WILL BE ASKED. IF THE ANSWER IS NO, -B- PORT WILL BE TESTED AND QUESTION 3 WILL BE ASKED.

2. 1ST 4K BANK #

THIS NUMBER THAT IS TYPED WILL DETERMINE WHERE THE BUFFER AREA WILL START ON -A- PORT. USE TABLE ABOVE

NOTE:

PROGRAM IS LOCATED IN 1ST 4K BANK. THEREFORE, THIS BANK CAN NOT BE USED AS A BUFFER.

283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330

AC-9318F-MC CZRSCF RH11-R503/LA-R504 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.

331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347

AC-9318F-MC CZRSCF RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 10
DESCRIPTION

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

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

AC-9318F-MC CZRSCF 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-9318F-MC CZRSCF RH11-RS03/LA-RS04 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

AC-9318F-MC CZRSCF 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 SWO 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

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-9318F-MC CZRSCF RH11-RS03/LA-RS04 DATA RELIABILITY DIAG PAGE 14
 DESCRIPTION

8.3 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 CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 ;COPYRIGHT 1973,1974,1975,1976,1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
 ;PROGRAM BY STANLEY HARACKIEWICZ

556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
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
014032
000052
040000
000174
000000
000176
000000
000200
000137 001230
000210
012706 000500
000214 000137 003154
    
```

```

; SWITCH USE
; -----
; SW15= 100000 ; HALT ON ERROR
; SW14= 40000 ; LOOP ON FUNCTION
; SW13= 20000 ; INHIBIT ERROR TIMEOUTS
; 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 TIMEOUT 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
; ; TRAP CATCHER FROM 0 - 776
; ; HOOKS FOR ACT 11

.= 0
.= 46
$ENDAD
.= 52
BIT14

.= 174 ; SOFTWARE SWITCK 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	002330		JMP	@#A1	; DATA BUFFER QUESTIONS
616							
617	000230	000137	015116		JMP	@#RLDR	; RESTORE LOADER
618							
619	000234	000137	003246		JMP	@#ADTL	; TRACK AND SECTOR SELECT TEST
620							; WRITE EACH WORD ADDR ON ITSELF AND READ IT BACK
621	000240	000137	004776		JMP	@#RANEL	; RANDOM ADDRESS, DATA TEST
622	000244	000137	012430		JMP	@#PFT1	; DISK WRITE POWER FAIL TEST
623	000250	000137	012756		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 ;CLEAR ALL REGISTERS

644					.CLR DV:	MOV	#40, @RSCS2	;CLEAR ALL REG
645							UNNUM, @RSCS2	;GET UNIT #
646	000316	012777	000040	000510				
647	000324	013777	001160	000502				
648	000332	000002				RTI		
649								

```

        .SBTTL          SKMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS
650
651
652          177572          SR0=177572          ;ADDRESS OF MEM MGMT REGISTER SR0
653          177574          SR1=177574          ;
654          177576          SR2=177576          ;
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

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCF.P11 16-DEC-77 15:10

MACY11 30(1046) 20-DEC-77 15:36 PAGE 19
SKMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0018

692				
693	000001	N=	1	; INITIALIZE FOR NEWTST
694	104000	HLT=	EMT	; SET HLT TO EMT FOR ERROR TYPEOUTS
695	177776	PS=	177776	; PROCESSOR STATUS
696	177776	PSW=	PS	; PROCESSOR STATUS WORD
697	000007	BELL=	7	; BELL
698	000000	R0=	%0	; R0 - DEFINE REGISTERS
699	000001	R1=	%1	; R1
700	000002	R2=	%2	; R2
701	000003	R3=	%3	; R3
702	000004	R4=	%4	; R4
703	000005	R5=	%5	; R5
704	000006	SP=	%6	; R6 - STACK POINTER
705	000007	PC=	%7	; R7 - PROGRAM COUNTER
706	000001	BIT0=	1	; BIT EQUATES
707	000002	BIT1=	2	
708	000004	BIT2=	4	
709	000010	BIT3=	10	
710	000020	BIT4=	20	
711	000040	BIT5=	40	
712	000100	BIT6=	100	
713	000200	BIT7=	200	
714	000400	BIT8=	400	
715	001000	BIT9=	1000	
716	002000	BIT10=	2000	
717	004000	BIT11=	4000	
718	010000	BIT12=	10000	
719	020000	BIT13=	20000	
720	040000	BIT14=	40000	
721	100000	BIT15=	100000	
722	000001	GOOD=	R1	; FOR GOOD DATA
723	000000	BAD=	R0	; FOR BAD DATA

724		000510			= 510	
725	000510	005015	047125	052111	LOADSW: .ASCIZ	<15><12>"UNIT # "
726	000516	021440	000040			
727	000522	005015	040506	040524	UNRECO: .ASCIZ	<15><12>"FATAL"<15><12><12>
728	000530	006514	005012	000		
729						
730						
731	000535	015	047412	052120	.OPDR: .ASCIZ	<15><12>"OPT DSK ADDR"
732	000542	042040	045523	040440		
733	000550	042104	000122			
734	000554	005015	040504	040524	DATA: .ASCIZ	<15><12>"DATA "
735	000562	000040				
736						
737	000564	051127	020124	051105	WRTERR: .ASCIZ	"WRT ERR"
738	000572	000122				
739						
740	000574	051127	020124	045503	WCKERR: .ASCIZ	"WRT CK ERR"
741	000602	042440	051122	000		
742						
743	000607	122	020104	051105	RDERR: .ASCIZ	"RD ERR"
744	000614	000122				
745						
746	000616	005015	042522	020103	RECOV: .ASCIZ	<15><12>"REC RETRY CT "
747	000624	042522	051124	020131		
748	000632	052103	000040			
749						
750	000636	005015	000		CRLFLF: .ASCIZ	<15><12>
751						
752	000641	040	054450	047440	YORN: .ASCIZ	" (Y OR N)"
753	000646	020122	024516	000		
754						
755	000653	015	030412	052123	STABUF: .ASCIZ	<15><12>"1ST 4K BANK # "
756	000660	032040	020113	040502		
757	000666	045516	021440	000040		
758						
759	000674	005015	040522	042116	RANDM: .ASCIZ	<15><12>"RANDOM "
760	000702	046517	000040			
761	000706	005015	042524	052123	TSTNG: .ASCIZ	<15><12>"TESTING "
762	000714	047111	020107	000		
763	000721	015	050012	053517	PDOWN: .ASCIZ	<15><12>"POWER DOWN"
764	000726	051105	042040	053517		
765	000734	000116				
766	000736	005015	040520	044522	PAREER: .ASCIZ	<15><12>"PARITY ERR"
767	000744	054524	042440	051122		
768	000752	000				
769						
770		000754			.EVEN	
771						

H02

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 CZRSCF.P11 16-DEC-77 15:10

MACY11 30(1046) 20-DEC-77 15:36 PAGE 21

\$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0020

```

772      001000      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 000000      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      TK: 177560
782 001022 177562      TKB: 177562
783 001024 177566      TPB: 177566      ;OUTPUT BUFFER
784 001026 177570      SWR: 177570      ;SWITCH REGISTER
785 001030 177570      DISPLAY: 177570      ;DISPLAY REGISTER
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      RSOB: 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
  
```

817 000006
 818 000000
 819 000250
 820 001070 000000
 821 001072 000000
 822 001074 000000
 823 001076 000000
 824 001100 000000
 825 001102 000000
 826 001104 000000
 827 001106 000000
 828 001110 000000
 829 001112 000000
 830 001114 000000
 831 001116 000000
 832 001120 000000
 833 001122 000000
 834 001124 000000

RW=6
 UP=0
 MMVEC=250
 STAMEM: 0
 SAVAST: 0
 STBCOM: 0
 SAVCPU: 0
 SAVMGA: 0
 SAVMGB: 0
 SAVMGC: 0
 SIZEAP: 0
 SIZEBP: 0
 WDCTB: 0
 AOB1: 0
 VADDR: 0
 PHADDR: 0
 FLAG2: 0
 DROP: 0

:R/W IN PDR REG
 :UP BITY IN PDR REG
 :ADDR OF MEM MGMT ERROR TRAP
 :STARTING LOC FOR -A- PORT
 :SAVE LOC FOR STAMEM
 :STARTING LOC FOR -B- PORT
 :SAVE LOC FOR CPUBM
 :STARTING ADDR FOR -A- PORT WITH MEM MGMT
 :STARTING ADDR FOR B PORT W/MEM MGMT
 :STARTING LOC FOR CPU W/MEM MGMT
 :SIZE OF A PORT
 :SIZE OF B PORT
 :WC FOR A PORT
 :FLAG FOR PORT BEING TESTED
 :VIRTUAL ADDR
 :PHYSICAL ADDR
 :FLAG FOR RESTART AND FOUND DRIVE
 :BAD UNITS ON SYSTEM THAT GET DUMPED

;DISCRIPTION OF FLAG2

835
 836
 837
 838
 839
 840
 841
 842
 843
 844
 845
 846
 847
 848
 849
 850
 851
 852
 853
 854
 855
 856
 857
 858
 859
 860
 861
 862
 863
 864
 865
 866
 867
 868
 869
 870

: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

```

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 HRDR: 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

```

909	001230	005037	020116		SETSWI:	CLR	SWI	
910	001234	012706	000500		BEGIN:	MOV	#500, SP	; SET STACK TO *** 500 ***
911	001240	012737	016216			MOV	#.POWER, @#24	; SET UP PF VECTOR
912	001246	012737	000340	000024		MOV	#340, @#26	; LOCK OUT THE WORLD
913	001254	012737	015660	000030		MOV	#.HLT, @#30	; SET EMT VECTOR
914	001262	012737	000340	000032		MOV	#340, @#32	; LOCK UP
915	001270	012737	016564	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	032737	000001	020116		BIT	#BIT0, SWI	
920	001322	001001				BNE	2\$	
921	001324	104444				SUSWR		; SIZE FOR SWITCHLESS
922	001326	042737	177677	001126	2\$:	BIC	#177677, FLAG	; CLEAR FLAG
923	001334	042737	177776	001122		BIC	#177776, FLAG2	; CLEAR ALL EXECPT RESTART
924	001342	005037	001216			CLR	FLAG3	; CLEAR LOOP IN ADDRESS + RANDOM TST FLAG
925	001346	032737	000001	001122		BIT	#BIT0, FLAG2	; IS THIS THE FIRST TIME?
926	001354	001002				BNE	1\$; NO
927	001356	004737	020120			JSR	PC, LDR	; SAVE LOADER
928	001362	000005			1\$:	RESET		; CLEAR THE WORLD
929	001364	012737	000340	177776		MOV	#340, PS	; LOCK UP INTERRUPT LEVELS
930	001372	004537	012374			JSR	R5, ERRCL	; CLEAR ERROR COUNTER + PASS CNT
931	001376	005037	001212			CLR	MMAVA	; CLEAR MEM MGMT FLAG
932	001402	005037	001114			CLR	AOB1	; TEST A PORT FIRST
933	001406	032777	000004	177412		BIT	#BIT2, @SWR	; WANT MEM MGMT?
934	001414	001021				BNE	3\$; NO
935	001416	012737	001444	000004		MOV	#5\$ 4	; SET TIMEOUT TRAP
936	001424	012737	000340	000006		MOV	#340, 6	; SET PS
937	001432	005037	177572			CLR	@#SR0	; IS MEM MGMT AVAILABLE?
938	001436	005137	001212			COM	MMAVA	; YES
939	001442	000401				BR	4\$; CONT
940	001444	022626			5\$:	CMP	(6)+, (6)+	; CLEAR STACK
941	001446	012737	000006	000004	4\$:	MOV	#6, 4	; RESET
942	001454	005037	000006			CLR	6	; TRAP
943	001460	032737	000001	001122	3\$:	BIT	#BIT0, FLAG2	; IS THIS THE FIRST TIME
944	001466	001002				BNE	CALM	; NO
945	001470	000137	020210			JMP	SIZZAP	; SIZE BUFFERS
946	001474	004737	011420		CALM:	JSR	PC, @#EXTMEM	; SET UP DATA BUFFERS
947	001500	004737	015144		CALM1:	JSR	PC, MAMK	; TURN ON PARITY MEM
948	001504	032737	000001	001122		BIT	#BIT0, FLAG2	; 1ST TIME ?
949	001512	001006				BNE	3\$; NO
950	001514	013737	001144	001214		MOV	SWRDCT, SAVWC	; SAVE WC FOR CONVERSATION MODE COMPARE
951	001522	013737	001112	001220		MOV	WDCTB, SAVWCB	; SAVE WC FOR -B- PORT
952	001530	052737	000001	001122	3\$:	BIS	#BIT0, FLAG2	; SET 1ST TIME FLAG
953	001536	005037	001134			CLR	DMA	; CLEAR DAR REGISTERS
954	001542	005037	001136			CLR	PATNU	; CLEAR PATTEN COUNT
955	001546	013737	001144	001130		MOV	SWRDCT, WRDCT	
956	001554	032777	000002	177244		BIT	#BIT1, @SWR	; DATA TEST ONLY?
957	001562	001403				BEQ	2\$; NO
958	001564	052737	002000	001126		BIS	#BIT10, FLAG	; YES
959	001572	032777	002000	177226	2\$:	BIT	#BIT10, @SWR	; ENTER CONVERSATION MODE?
960	001600	001007				BNE	1\$; YES GO TO CONVERSATION MODE
961	001602	052737	074000	001126		BIS	#74000, FLAG	
962	001610	004537	010142			JSR	R5, RESTOR	; RESTORE ORIGINAL WD CNT
963	001614	000137	003154			JMP	ADTST	
964	001620	000137	002220		1\$:	JMP	@#CONM	; ENTER CONVERSATION MODE


```

965 ;FIND OUT HOW MANY DRIVES
966 ;FIRST TEST RSAS
967
968 001624 012701 000010 DRVENO: MOV #8,R1 ;PUT 8 INTO R1 FOR COUNT
969 001630 042737 000002 001122 BIC #BIT1,FLAG2 ;CLEAR FOUND DRIVE FLG
970 001636 012777 000000 177170 MOV #0,RS2 ;SET DEVICE TO ZERO
971 001644 012777 000007 177174 TRY: MOV #7,RSER ;CAUSE AN ERROR +SETS BIT IN AS REG
972 001652 005301 DEC R1 ;DO A MAXIMUM OF 16 TIMES
973 001654 001403 BEQ DVNUM ;TESTED FOR ALL DRIVES GET OUT
974 001656 005277 177152 INC RS2 ;INCREMENT DRIVE UNIT
975 001662 000770 BR TRY ;REPEAT FOR NEXT DRIVE
976 001664 017737 177160 001162 DVNUM: MOV RSAS,UNITSV ;SAVE
977 001672 043737 001124 001162 BIC DROP,UNITSV ;DROP BAD DRIVES
978 001700 012737 000401 001164 MOV #401,UNCOMP ;SETUP TO CMP WITH UNITSV
979 001706 012737 000000 001160 MOV #0,UNNUM ;PUT 0 INTO UNIT NO.
980 001714 032777 000040 177104 BIT #BITS,RSW ;INHIBIT TYPE OUT?
981 001722 001005 BNE STTEST ;YES
982 001724 104402 000706 TYPE TSTNG
983 001730 042737 000004 001122 BIC #BIT2,FLAG2 ;CLEAR ERROR FLAG
984 001736 033737 001164 001162 STTEST: BIT UNCOMP,UNITSV ;IS THIS DRIVE ON THE SYSTEM
985 001744 001473 BEQ TRYNX ;NO
986 001746 013777 001160 177060 UNTYP: MOV UNNUM,RS2 ;YES PUT UNIT # INTO CS2
987 001754 005037 001166 CLR RS4DT ;CLEAR DRIVE TYPE FLAG
988 001760 022777 000004 177072 CMP #4,RSDT ;RS03LA?
989 001766 001004 BNE B$ ;NO
990 001770 012737 000004 001166 MOV #4,RS4DT ;SET DRIVE TYPE FLAG
991 001776 000422 BR 1$ ;CONT
992 002000 005777 177054 B$: TST RSDT ;IS THIS A RS03?
993 002004 001417 BEQ 1$ ;YES
994 002006 022777 000001 177044 2$: CMP #1,RSDT ;IS THIS A RS03 4US?
995 002014 001413 BEQ 1$ ;YES
996 002016 022777 000002 177034 3$: CMP #2,RSDT ;IS THIS A RS04?
997 002024 001404 BEQ 6$ ;YES
998 002026 022777 000003 177024 CMP #3,RSDT ;RS04?
999 002034 001037 BNE TRYNX ;GET A NEW NUMBER
1000 002036 052737 177777 001166 6$: BIS #-1,RS4DT ;YES RS04
1001 002044 032737 040000 001122 1$: BIT #BIT14,FLAG2 ;IN POWER FAIL OR CONVERSATION?
1002 002052 001401 BEQ 7$ ;NO
1003 002054 000207 RTS PC ;YES
1004 002056 032777 000200 176760 7$: BIT #BIT7,RS0S ;IS THIS DRIVE READY ?
1005 002064 001423 BEQ TRYNX ;NO GET ANOTHER DRIVE
1006 002066 032777 000040 176732 BIT #BITS,RSW ;TYPEOUT?
1007 002074 001016 BNE 4$ ;NO
1008 002076 032737 000004 001122 BIT #BIT2,FLAG2 ;WAS THERE AN ERRER?
1009 002104 001402 BEQ 5$ ;NO
1010 002106 104402 000636 TYPE ,CRLF
1011 002112 5$:
1012 002112 013746 001160 MOV UNNUM,-(6) ;PUT UNNUM ON STACK
1013 002116 104406 TYPES TYPE STACK IN OCTAL - SUPRESS
1014 002120 104402 000040 TYPE 40 ;TYPE SPACE
1015 002124 042737 000004 001122 BIC #BIT2,FLAG2 ;CLEAR ERROR FLAG
1016 002132 000426 4$: BR NOWGO ;NOW TEST

```

M02

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST

MACY11 30(1046) 20-DEC-77 15:36 PAGE 26

CZRSCF.P11 16-DEC-77 15:10

\$KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

SEQ 0025

1017	002134	006337	001164			TRYNX: ASL	UNCMP		;CHECK NEXT BIT FOR DRIVE
1018	002140	103403				BCS	CHCKDV		;DID WE TEST ANY REG?
1019	002142	005237	001160			INC	UNNUM		;INC UNIT #
1020	002146	000673				BR	STTEST		;CHECK FOR NEXT DRIVE
1021									
1022									;THIS PROGRAM WILL DEFAULT TO TESTING UNIT 0 IF IT CAN NOT FIND ANY DRIVES
1023	002150	032737	000002	001122		CHCKDV: BIT	#BIT1,FLAG2		;FOUND DRIVE?
1024	002156	001012				BNE	DONEE		;YES WE DID TEST A DRIVE
1025	002160	012737	100000	001164		MOV	#100000,UNCMP		;NO DRIVES TESTED, COULD NOT SET
1026	002166	005037	001160			CLR	UNNUM		;ANY AS BITS, THUS DEFAULTS TO 0
1027	002172	013746	001160			MOV	UNNUM,-(6)		;PUT UNNUM ON STACK
1028	002176	104406				TYPES			;TYPE STACK IN OCTAL - SUPPRESS
1029	002200	000000				HALT			;COULD NOT SET ANY ATA BITS
1030									;BY SETTING ERROR BITS
1031									;GO BACK AND USE OTHER DIAG.
1032	002202	000402				BR	NOWGO		;DEFAULT TO DRIVE 0
1033	002204	000137	013410			JMP	OUT		;GET OUT
1034	002210	052737	000002	001122		NOWGO: BIS	#BIT1,FLAG2		;FOUND DRIVE
1035	002216	000207				RTS	PC		
1036									
1037									;ENTER OPERATOR CONVERSATION MODE
1038									
1039	002220					CONM:			
1040	002220	104402	002224			TYPE	+2		;ASCIZ <15><12>"-A- PORT"
1041	002240	004737	003230			JSR	PC,CMPY		;COMPARE FOR YES
1042	002244	001405				BEQ	2\$;YES
1043	002246	012737	177777	001114		MOV	#-1,AOB1		;B PORT
1044	002254	000137	002276			JMP	1\$;TEST -B- PORT
1045	002260	104402	000653			2\$: TYPE	,STABUF		
1046	002264	104420				RDOCT			
1047	002266	012637	001070			MOV	(6)+,STAMEM		;START BUFFER AT 4K
1048	002272	000137	002316			JMP	NOPORT		;GET OUT

1049	002276	104402	000653		1\$:	TYPE	,STABUF	
1050	002302	104420				RDOCT		:GET ANS
1051	002304	012637	001074			MOV	(6)+,STBCOM	:AND SAVE IT
1052	002310	052737	000100	001126		BIS	#BIT6,FLAG	:SET B PORT FLAG
1053	002316	004737	011420		NOPORT:	JSR	PC,EXTMEM	:CAL BUFFERS AND WC
1054	002322	013737	001144	001130		MOV	SWORDCT,WRDCT	:GET STANDARD WC
1055	002330	052737	002000	001126	A1:	BIS	#BIT10,FLAG	:SET BIT FOR DATA TEST ONLY
1056	002336	004537	012374			JSR	RS,ERRCL	:CLEAR ERROR CNT + PASS CNT
1057	002342	042737	174040	001126		BIC	#174040,FLAG	:CLEAR MULTI FLAG MODE +PATTERN SELECT
1058	002350	104402	002354			TYPE	.+2	:ASCIZ <15><12>"MULTI DRIVE"
1059	002372	004737	003230			JSR	PC,CMPY	:COMPARE FOR YES
1060	002376	001004				BNE	DATTES	:ANS IS NO
1061	002400	052737	004000	001126		BIS	#BIT11,FLAG	:SET BIT FOR MULTI DRIVE
1062	002406	000434			1\$:	BR	ASKWC	
1063	002410	104402	000510		DATTES:	TYPE	,LOADSW	
1064	002414	104420				RDOCT		
1065	002416	012637	001170			MOV	(6)+,NUMS	:GET NUMBER
1066	002422	022737	000010	001170		CMP	#10,NUMS	:CORRECT # ?
1067	002430	103767				BLO	DATTES	:NO
1068	002432	013737	001170	001160		MOV	NUMS,UNNUM	:SET UNIT #
1069	002440	004737	006466			JSR	PC,FNDTYP	:TEST FOR R504 OR 03
1070	002444	005002			1\$:	CLR	R2	:CLEAR WORK AREA
1071	002446	000261				SEC		:SET CARRY
1072	002450	006102			2\$:	ROL	R2	:SET BIT IN WORK
1073	002452	005737	001170			TST	NUMS	:IS THIS THE RIGHT BIT FOR THE RIGHT DISK
1074	002456	001403				BEQ	3\$:YES
1075	002460	005337	001170			DEC	NUMS	:NO TRY AGAIN
1076	002464	000771				BR	2\$:TEST AGAIN
1077	002466	010237	001162		3\$:	MOV	R2,UNITSV	:SET DRIVE BIT IN UNITSV
1078	002472	052737	000002	001122		BIS	#BIT1,FLAG2	:SET FOUND DRIVE FLAG
1079								
1080	002500				ASKWC:	TYPE	.+2	:ASCIZ <15><12>"OPT WD CT"
1081	002500	104402	002504			JSR	PC,CMPY	:COMPARE FOR YES
1082	002520	004737	003230			BEQ	WCCON	:YES
1083	002524	001401				BR	OPDAR	:CONT
1084	002526	000444						

Address	Offset	Target	Source	Label	Operation	Comment
1085	002530				TYPE	;.ASCIZ <15><12>"WD CT "
1086	002530	104402	002534		RDOCT	;.ASCIZ <15><12>"WD CT "
1087	002546	104420			MOV	(6)+,NUMS ;GET NUMBER
1088	002550	012637	001170		TST	NUMS ;IS IT 0?
1089	002554	005737	001170		BEQ	WCCON ;YES ASK AGAIN FOR LENGTH
1090	002560	001763			MOV	SAVWC,R2 ;GET STANDARD WC FOR -A- PORT
1091	002562	013702	001214		TST	AOB1 ; -A- PORT?
1092	002566	005737	001114		BEQ	1\$;YES
1093	002572	001402			MOV	SAVWC,R2 ;GET WC FOR -B- PORT
1094	002574	013702	001220		INC	R2
1095	002600	005202		1\$:	CMP	R2,NUMS ;IS NUMS LESS THAN SWRDCT
1096	002602	020237	001170		BLOS	WCCON ;YES ASK FOR COUNT AGAIN
1097	002606	101750			MOV	NUMS,SWRDCT ;OPERATING WORD COUNT
1098	002610	013737	001170	001144	MOV	SWRDCT,WRDCT
1099	002616	013737	001144	001130	TST	AOB1 ;B PORT?
1100	002624	005737	001114		BEQ	OPDAR ;NO
1101	002630	001403			MOV	WRDCT,WDCRB ;YES GET WC
1102	002632	013737	001130	001112		
1103						
1104	002640	104402	000535		OPDAR: TYPE	.OPDR
1105	002644	004737	003230		JSR	PC,CMPLY ;COMPARE FOR YES
1106	002650	001037			BNE	OPPAT ;ANS IS NO
1107	002652	052737	000040	001126	BIS	#BITS,FLAG ;SET OPTIONAL DMA FLAG
1108	002660	104402	002664		TYPE	;.ASCIZ "="
1109	002666	104420			RDOCT	;.ASCIZ "="
1110	002670	012637	001170		MOV	(6)+,NUMS ;GET NUMBER
1111	002674	022737	000004	001166	CMP	#4,RS04DT ;RS03LA?
1112	002702	001004			BNE	3\$;NO
1113	002704	022737	003777	001170	CMP	#3777,NUMS ;IS ADDR. CORRECT?
1114	002712	000412			BR	2\$
1115	002714	005737	001166		3\$:	TST RS04DT ;IS THIS A RS04?
1116	002720	001404			BEQ	1\$;NO
1117	002722	022737	017777	001170	CMP	#17777,NUMS ;IS ADD. CORRECT?
1118	002730	000403			BR	2\$;GET OUT
1119	002732	022737	007777	001170	1\$:	CMP #7777,NUMS ;IS ADD. CORRECT?
1120	002740	101737			2\$:	BLOS OPDAR ;NO
1121	002742	013737	001170	001142	MOV	NUMS,TDMA ;TEMP SECTOR REGISTER

1122	002750				OPPAT:				
1123	002750	104402	002754			TYPE	,.+2		; .ASCIZ <15><12>"PATN # "
1124	002766	104420				RDOCT			
1125	002770	012637	001170			MOV	(6)+, NUMS		; GET NUMBER
1126	002774	022737	000023	001170		CMP	#23, NUMS		; TEST FOR CORRECT NO
1127	003002	101762				BLOS	OPPAT		; ASK AGAIN
1128	003004	005037	001136			CLR	PATNU		; CLEAR PATTERN #
1129	003010	022737	000022	001170		CMP	#22, NUMS		
1130	003016	001411				BEQ	OPWRT		; DATA PATTERN UNDER PROGRAM CONTROL
1131	003020	052737	100000	001126		BIS	#BIT15, FLAG		; SET PROGRAM FLAG
1132	003026	013737	001170	001136		MOV	NUMS, PATNU		; OPERATOR WANTS TO SELECT DATA
1133	003034	000241				CLC			
1134	003036	006137	001136			ROL	PATNU		
1135									
1136	003042				OPWRT:				
1137	003042	104402	003046			TYPE	,.+2		; .ASCIZ <15><12>"WRITE"
1138	003056	004737	003230			JSR	PC, CMPY		; COMPARE FOR YES
1139	003062	001003				BNE	OPRD		; ASK ABOUT WRITE CHECK
1140	003064	052737	040000	001126		BIS	#BIT14, FLAG		; YES SET FLAG BIT
1141									
1142	003072				OPRD:				
1143	003072	104402	003076			TYPE	,.+2		; .ASCIZ <15><12>"READ"
1144	003106	004737	003230			JSR	PC, CMPY		; COMPARE FOR YES
1145	003112	001003				BNE	OPWCK		
1146	003114	052737	010000	001126		BIS	#BIT12, FLAG		; SET FLAG TO READ
1147									
1148	003122				OPWCK:				
1149	003122	104402	003126			TYPE	,.+2		; .ASCIZ <15><12>"WRT CK"
1150	003140	004737	003230			JSR	PC, CMPY		; COMPARE FOR YES
1151	003144	001003				BNE	ADTST		
1152	003146	052737	020000	001126		BIS	#BIT13, FLAG		
1153									
1154	003154	032737	004000	001126	ADTST:	BIT	#BIT11, FLAG		; ARE WE IN MULTI DRIVE MODE
1155	003162	001402				BEQ	EXMFLG		; BRANCH IF NO.
1156	003164	004737	001624			JSR	PC, DRVENO		; GET DRIVES TO BE TESTED
1157	003170	042737	000004	001126	EXMFLG:	BIC	#BIT2, FLAG		; CLEAR XFER MODE FLAG
1158	003176	032777	000400	175622		BIT	#BIT8, JSWR		; XFER MODE?
1159	003204	001403				BEQ	1\$; NO
1160	003206	052737	000004	001126		BIS	#BIT2, FLAG		; SET XFER MODE FLAG
1161	003214	032737	002000	001126	1\$:	BIT	#BIT10, FLAG		; TEST FOR DATA TEST ONLY
1162	003222	001414				BEQ	ADT1		; DO COMPLETE TEST
1163	003224	000137	003640			JMP	DATAT		; DO DATA TEST ONLY
1164									
1165	003230	104402	000641		CMPY:	TYPE	, YORN		
1166	003234	104422				RDLIN			
1167	003236	122737	000131	016774		CMPB	#'Y, INPUT		; TEST FOR YES
1168	003244	000207				RTS	PC		
1169	003246	052737	100000	001216	ADTL:	BIS	#BIT15, FLAG3		; SET LOOP IN ADDRESS TEST FLAG GOT HERE
1170									; BECAUSE PROGRAM WAS STARTED AT 234

```

1171 ;RH11 ADDRESS TEST #1 (TRACK AND SECTOR SELECTION TEST)
1172 ;WRITE 100(OCTAL) RS03, 200(OCTAL) RS04, WORDS IN EACH SECTOR
1173 ;THE WORD CONTAINS THE ADDRESS OF EACH SECTOR
1174 ;WHEN THE COMPLETE DISK IS WRITTEN READ
1175 ;BACK EACH SECTOR AND COMPARE FOR THE CORRECT
1176 ;DATA IN THE SECTOR
1177 ;PS IS AT LEVEL 7 SO NO INTERRUPTS
1178
1179 003254 ADT1: ;ADDRESS TEST
1180 ;*****
1181 ;TEST 1 ADDRESS TEST
1182 ;*****
1183 003254 104400 TST1: SCOPE
1184 003256 032737 000004 001126 ADT1A: BIT #BIT2,FLAG ;XFER MODE?
1185 003264 001402 BEQ 3$ ;NO
1186 003266 000137 003640 JMP DATAT ;YES
1187 003272 012737 000340 177776 3$: MOV #340,PS ;LOCK UP PS
1188 003300 012737 020000 017360 MOV #20000,OUTBUF ;START BUF AT 20000
1189 003306 052737 000400 001126 BIS #BIT8,FLAG ;SET TEST FLAG
1190 003314 013737 001144 001150 MOV SWRDCT,SAVE ;SAVE STD WD COUNT
1191 003322 005037 001134 CLR DMA ;CLEAR DISK ADD
1192 003326 104426 CLRDV ;INIT DRIVE
1193 003330 004737 006510 JSR PC,WHTHU ;GET WORD COUNT
1194 003334 013737 001130 001144 5$: MOV WRDCT,SWRDCT
1195 003342 013737 017360 001140 2$: MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
1196 003350 104414 SEABUF: ERCLR ;CLEAR RS REGISTERS IF ERROR
1197 003352 013700 017360 MOV OUTBUF,R0 ;SET UP ADDRESS BUFFER
1198 003356 013701 001130 MOV WRDCT,R1
1199 003362 013720 001134 XSEABUF: MOV DMA,(0)+ ;LOAD OUTBUF WITH DATA TO BE WRITTEN
1200 003366 005301 DEC R1 ;FILL OUTBUF
1201 003370 001374 BNE XSEABUF ;WITH DATA
1202 003372 012737 000061 001172 MOV #61,CMD ;WRITE NO I/E
1203 003400 104416 DKCMD ;GO WRITE
1204 003402 105777 175424 TSTB @RSCS1 ;CHECK FOR READY
1205 003406 100375 BPL -4
1206 003410 005777 175416 TST @RSCS1 ;TEST FOR ERROR
1207 003414 100010 BPL WRNEXB ;BRANCH IF NO ERROR
1208 003416 012737 003350 001010 MOV #SEABUF,LAD ;SET UP LOOP ADDRESS
1209 003424 052737 001000 001126 BIS #BIT9,FLAG ;SET ERROR BIT IN FLAG
1210 003432 104430 LOGW ;LOG WRITE ERROR
1211 003434 104034 HLT !WC!DA!BA
1212 003436 104400 WRNEXB: SCOPE
1213 003440 004737 006776 JSR PC,DISBUF ;SET UP NEXT DISK ADDR.
1214 003444 000741 BR SEABUF ;WRITE NEXT SECTOR
1215 003446 104400 RRDSEC: SCOPE

```

E03

CZRSCF RH11-RS03-RS03/LA-RS04
CZRSCF.P11 16-DEC-77 15:10

DATA AND RELIABILITY TEST
TST1 ADDRESS TEST

MACY11 30(1046) 20-DEC-77 15:36 PAGE 31

SEQ 0030

```

1216 003450 104414
1217 003452 012737 000071 001172
1218 003460 104416
1219 003462 105777 175344
1220 003466 100375
1221 003470 005777 175336
1222 003474 100006
1223 003476 052737 001000 001126
1224 003504 104432
1225 003506 104014
1226 003510 104400
1227 003512 013702 017360
1228 003516 013746 001130
1229 003522 004737 006510
1230 003526 013703 001130
1231 003532 012637 001130
1232 003536 023712 001134
1233 003542 001004
1234 003544 005722
1235 003546 005303
1236 003550 001372
1237 003552 000412
1238 003554 013701 001134
1239 003560 011200
1240 003562 104000
1241 003564 104436
1242 003566 004737 014146
1243 003572 001002
1244 003574 104402 000636
1245
1246
1247
1248 003600 104400
1249 003602 004737 006776
1250 003606 000717
1251 003610 013737 001150 001144
1252 003616 042737 000400 001126
1253 003624 032737 100000 001216
1254 003632 001402
1255 003634 000137 003256

RDSECT: ERCLR ;CLEAR ERRORS
MOV #71,CMD ;READ NO I/E
DKCMD ;DO A READ
TSTB @RSCS1 ;CHECK FOR READY
BPL -4 ;NOT READY BRANCH BACK
TST @PSCS1 ;TEST FOR ERROR
BPL ADHGT ;BRANCH IF NO ERROR
BIS #BIT9,FLAG ;SET ERROR FLAG
LOGR ;LOG READ ERROR
HLT !WC!DA

ADHGT: MOV OUTBUF,R2
MOV WRDCT, -(SP) ;SAVE OLD WC
JSR PC,WHTHU
MOV WRDCT,R3
MOV (SP)+,WRDCT ;RESTORE OLD WC
SANHT: CMP DMA,(2) ;CMP FOR CORRECT ADDR.
BNE ADERR ;BRANCH IF DATA DID NOT COMPARE
TST (2)+ ;GET NEXT ADDRESS OF INBUF
DEC R3 ;DEC SECTOR COUNT
BNE SANHT ;TEST NEXT WORD
BR CHKADT
ADERR: MOV DMA,GOOD ;CORRECT ADDRESS
MOV (2),BAD ;DATA IN ERROR
HLT ;DISK ADD DID NOT MATCH WRITTEN ADDRESS
LOGC ;LOG COMPARE ERROR
JSR PC,PRNT ;INHIBIT TYPEOUT?
BNE CHKADT ;YES
TYPE ,CRLF

;*****REPORT ONLY ONE ERROR PER SECTOR*****

CHKADT: SCOPE
JSR PC,DISBUF ;SET UP NEXT DISK BUFFER
BR RRDSEC ;CHECK NEXT SECTOR
MOV SAVE,SWRDCT ;GET STD WD COUNT
BIC #BIT0,FLAG ;CLEAR TEST FLAG
BIT #BIT15,FLAG3 ;DOES OPERATOR WANT TO LOOP ON TEST
BEQ +6 ;NO
JMP ADT1A ;YES

```

```

1256 003640          DATAT:                               ;DATA TEST
1257                ;*****
1258                ;TEST 2          DATA TEST
1259                ;*****
1260 003640 104400    †ST2:  SCOPE
1261 003642 005037 001134    CLR      DMA          ;CLEAR DISK ADDRESS
1262 003646 104426    CLRDV   ;CLEAR RS REGISTERS
1263 003650 013737 001144 001130  MOV     SWRDCT,WRDCT
1264 003656 032737 000100 001126  BIT     #BIT6,FLAG
1265 003664 001403    BEQ     3$          ;MULTI PORT?
1266 003666 005737 001114    TST     AOB1         ;NO
1267 003672 001003    BNE     1$          ;A OR B PORT?
1268 003674 004737 011276    3$:  JSR     PC,APORT   ;B PORT
1269 003700 000402    BR      2$          ;A PORT
1270 003702 004737 011344    1$:  JSR     PC,BPORT  ;B PORT
1271 003706 004737 006722    2$:  JSR     PC,VECTRR ;SETUP INT VECTOR
1272 003712 012777 000340 175144  MOV     #340,STATUS ;SET DISK STATUS REG LOC (206)
1273 003720 012737 003752 001152  MOV     #LDAT,HRDR   ;SETUP FOR HARD ERROR RETURN
1274 003726 013737 017360 001140  MOV     OUTBUF,BUF   ;SETUP OUTPUT BUFFER
1275 003734 052737 000003 001126  BIS     #3,FLAG      ;SET COUNTER TO 3
1276 003742 004537 007644    JSR     RS,PASEL     ;SET UP DATA BUFFERS
1277 003746 005037 001200    LDAT1: CLR     LOPCNT ;CLEAR ERROR FLAG
1278 003752 104414    LDAT:  ERCLR        ;CLEAR RS REG. IF ERROR
1279 003754 004537 006426    JSR     RS,OPDSEL   ;SET UP DISK ADDRESS
1280 003760 032737 040000 001126  BIT     #BIT14,FLAG ;TEST FOR WRITE
1281 003766 001456    BEQ     SLH          ;TEST FOR READ
1282 003770 012737 000161 001172  MOV     #161,CMD     ;WRITE WITH I/E
1283 003776 104416    DKCMD   ;DO A WRITE
1284 004000 004737 011712    JSR     PC,WATT     ;WAIT FOR INTERRUPT
1285 004004 012737 003752 001010  MOV     #LDAT,LAD    ;SETUP SCOPE LOOP
1286 004012 032737 001000 001126  BIT     #BIT9,FLAG  ;WAS THERE AN ERROR?
1287 004020 001423    BEQ     WRXBL       ;CONT
1288 004022 104430    LOGW   ;LOG WRITE ERROR
1289 004024 005237 001200    INC     LOPCNT      ;SET ERROR FLAG
1290 004030 004737 014146    2$:  JSR     PC,PRNT   ;TYPE ?
1291 004034 001004    BNE     1$          ;NO
1292 004036 104402 000554    TYPE   ,DATA
1293 004042 104402 000564    TYPE   ,WRTERR
1294 004046 104044    1$:  HLT     !DS!DA    ;WRITE ERROR
1295 004050 005337 001126    DEC     FLAG        ;DEC COUNTER
1296 004054 032737 000003 001126  BIT     #3,FLAG     ;DONE YET WITH 3RD TRY?
1297 004062 001333    BNE     LDAT        ;NOT 3 TRIES YET? TRY AGAIN
1298 004064 004737 011402    JSR     PC,WTNO    ;TYPE CAN NOT WRITE

```


1299	004070	005737	001200		WRXBL:	TST	LOPCNT		; WAS THERE AN ERROR?
1300	004074	001402				BEQ	WRX1		; NO
1301	004076	004737	011734			JSR	PC, TYPREC		; TYPE RECOVERED
1302	004102	005037	001200		WRX1:	CLR	LOPCNT		; CLEAR ERROR FLAG
1303	004106	104400				SCOPE			
1304	004110	052737	000003	001126		BIS	#3, FLAG		; CLEAR RETRY COUNT
1305	004116	004737	006776			JSR	PC, DISBUF		; SET BUFFER FOR WRITE CHECK
1306	004122	000711				BR	LDAT1		
1307	004124	104400			SLH:	SCOPE			
1308	004126	104414			SLH2:	ERCLR			; CLEAR RS REG IF ERRORS
1309	004130	004537	006426			JSR	RS, OPDSEL		; IS THE OPERATOR SELECTING THE TRACK
1310	004134	032737	020000	001126		BIT	#BIT13, FLAG		; TEST FOR WRITE CHECK
1311	004142	001002				BNE	1\$; YES
1312	004144	000137	004450			JMP	ESH1		; NO
1313	004150	013737	017360	001140	1\$:	MOV	OUTBUF, BUF		; SET UP CURRENT ADDRESS
1314	004156	012737	000151	001172		MOV	#151, CMD		; WRITE CHECKWITH I/E
1315	004164	104416				DKCMD			; GO WRITE CHECK
1316	004166	004737	011712			JSR	PC, WATT		; WAIT FOR INTERRUPT
1317	004172	032737	001000	001126	XESH:	BIT	#BIT9, FLAG		; IS THERE AN ERROR?
1318	004200	001505				BEQ	1\$; NO ERROR
1319	004202	005737	001200			TST	LOPCNT		; 1ST ERROR?
1320	004206	001001				BNE	2\$; NO
1321	004210	104434				LOGWC			; YES LOG ERROR
1322	004212	032777	000100	174606	2\$:	BIT	#BIT6, JSWR		; TYPE ALL ERRORS?
1323	004220	001007				BNE	3\$; YES
1324	004222	032777	001000	174576		BIT	#BIT9, JSWR		; LOOP ON ERROR?
1325	004230	001003				BNE	3\$; YES
1326	004232	005737	001200			TST	LOPCNT		; FIRST ERROR?
1327	004236	001056				BNE	10\$; NO
1328	004240	004737	014146		3\$:	JSR	PC, PRNT		; TYPE OUT?
1329	004244	001052				BNE	4\$; NO
1330	004246	104402	000554			TYPE	, DATA		
1331	004252	104402	000574			TYPE	, WCKERR		
1332	004256	017702	174556			MOV	RSBA, R2		; GET CORRECT BA
1333	004262	023702	017360			CMP	OUTBUF, R2		; DID A WD GET XFERED?
1334	004266	001406				BEQ	9\$; NO
1335	004270	032777	000400	174530		BIT	#BIT8, JSWR		; XFER MODE?
1336	004276	001002				BNE	9\$; YES
1337	004300	162702	000002			SUB	#2, R2		
1338	004304	004737	014146		9\$:	JSR	PC, PRNT		; TYPEOUT ERRORS?
1339	004310	001030				BNE	4\$; NO
1340	004312	005737	001212			TST	MMAVA		; IS MEM MGMT AVAILABLE?
1341	004316	001402				BEQ	7\$; NO

H03

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
 CZRSCF.P11 16-DEC-77 15:10

MACY11 30(1046) 20-DEC-77 15:36 PAGE 34

SEQ 0033

Address	OpCode	Op1	Op2	Op3	Op4	Label	Instruction	Comments
1342	004320	004737	006554				JSR PC,PHYCOV	;YES GET VITURAL ADDR
1343	004324	010237	001222			7\$:	MOV R2,WORK	;GET BA
1344	004330					8\$:		
1345	004330	104402	004334				TYPE ,.+2	; .ASCIZ <15><12>"(BA)=""
1346	004344					6\$:		
1347	004344	017746	174652				MOV @WORK,-(6)	;PUT @WORK ON STACK
1348	004350	104404					TYPEO	;TYPE STACK IN OCTAL
1349	004352	104402	004356				TYPE ,.+2	; .ASCIZ "WC=""
1350	004364	017746	174446				MOV @RSWC,-(6)	;PUT @RSWC ON STACK
1351	004370	104404					TYPEO	;TYPE STACK IN OCTAL
1352	004372	104026				4\$:	HLT !DA!DB!BA	;NOTE: BA REG. = +2 OF ACTUAL MEMORY
1353								;LOC AFTER WORDS HAVE BEEN XFERED
1354	004374	005237	001200			10\$:	INC LOPCNT	;INC ERROR COUNT
1355	004400	022737	000010	001200			CMP #10,LOPCNT	;10 TRYS YET?
1356	004406	001247					BNE SLH2	;NO
1357	004410	004737	006446				JSR PC,NOREC	;TYPE UNRECOVERABLE
1358	004414	005737	001200			1\$:	TST LOPCNT	;ANY ERRORS?
1359	004420	001402					BEQ 5\$;NO
1360	004422	004737	011734				JSR PC,TYPREC	;TYPE RECOVERED
1361	004426	005037	001200			5\$:	CLR LOPCNT	;CLEAR ERROR COUNTER
1362	004432	104400					SCOPE	
1363	004434	012737	004126	001010			MOV #SLH2,LAD	;SETUP LOOP ADDRESS
1364	004442	004737	006776				JSR PC,DISBUF	;SET UP THE DISK BUFFER
1365	004446	000422					BR SLH2A	
1366	004450	004537	011230			ESH1:	JSR R5,CLEAR	;CLEAR BUFFER
1367	004454	004537	006426			ESH:	JSR R5,OPDSEL	;OPERATOR SELECTED DISK ADDRESS?
1368	004460	032737	010000	001126			BIT #BIT12,FLAG	;TEST FOR READ
1369	004466	001002					BNE 1\$;YES
1370	004470	000137	004702				JMP MSTR	;NO READ
1371	004474	104400				1\$:	SCOPE	
1372	004476	042737	000003	001126			BIC #3,FLAG	;CLEAR RE-READ COUNT
1373	004504	005037	001200				CLR LOPCNT	;CLEAR FLAG
1374	004510	000137	004520				JMP DSKRD	;CONT
1375	004514	000137	004126			SLH2A:	JMP SLH2	
1376	004520	104414				DSKRD:	ERCLR	;CLEAR RS REG IF ERRORS
1377	004522	012737	000171	001172			MOV #171,CMD	;READ WITH I/E
1378	004530	104416					DKCMD	;READ
1379	004532	004737	011712				JSR PC,WATT	;WAIT FOR INTERRUPT
1380	004536	032777	010000	174262			BIT #10000,@SWR	;COMPARE?
1381	004544	001006					BNE ELH	;NO
1382	004546	032737	000004	001126			BIT #BIT2,FLAG	;COMPARE?
1383	004554	001002					BNE ELH	;NO
1384	004556	004537	010374				JSR R5,COMPARE	;COMPARE

1385	004562	032737	001000	001126	ELH:	BIT	#BIT9,FLAG	; IS THERE AN ERROR?
1386	004570	001433				BEQ	ADR0	; NO
1387	004572	032777	000100	174226		BIT	#BIT6,ASWR	; SOFT ERROR TYPEOUT?
1388	004600	001003				BNE	3\$; YES
1389	004602	005737	001200			TST	LOPCNT	; FIRST ERROR?
1390	004606	001011				BNE	2\$; NO
1391	004610	004737	014146		3\$:	JSR	PC,FRNT	; TYPEOUT?
1392	004614	001004				BNE	1\$; NO
1393	004616	104402	000554			TYPE	,DATA	
1394	004622	104402	000607			TYPE	,RDERR	
1395	004626	104432			1\$:	LOGR		; LOG READ ERROR
1396	004630	104044				HLT	!DS!DA	
1397	004632	104400			2\$:	SCOPE		
1398	004634	005237	001200			INC	LOPCNT	; COUNT ERRORS
1399	004640	022737	000010	001200		CMP	#10,LOPCNT	; LAST RETRY?
1400	004646	001324				BNE	DSKRD	; NO
1401	004650	004737	006446			JSR	PC,NOREC	; TYPE UNRECOVERABLE
1402	004654	000137	004674		4\$:	JMP	ADR1	; CONT
1403	004660	104400			ADR0:	SCOPE		
1404	004662	005737	001200			TST	LOPCNT	; ANY ERRORS?
1405	004666	001402				BEQ	ADR1	; NO
1406	004670	004737	011734			JSR	PC,TYPREC	; TYPE RECOVERED
1407	004674	004737	006776		ADR1:	JSR	PC,DISBUF	; GO SET UP DISK BUFFER.
1408	004700	000434				BR	READCT	; CONT. READING
1409	004702	005737	001126		MSTR:	TST	FLAG	
1410	004706	100423				BMI	3\$; OPERATOR SELECTED PATTERN
1411	004710	062737	000002	001136		ADD	#2,PATNU	; INC PATTERN INDEX
1412	004716	022737	000044	001136		CMP	#4,PATNU	
1413	004724	001402				BEQ	+6	
1414	004726	000137	003640			JMP	DATAT	; NOT LAST PATTERN EXIT
1415	004732	005037	001136			CLR	PATNU	; LAST PATTERN EXIT
1416	004736	032777	000002	174062		BIT	#BIT1,ASWR	; DATA TEST ONLY?
1417	004744	001006				BNE	2\$; YES
1418	004746	032737	002000	001126		BIT	#BIT10,FLAG	; DATA TEST ONLY?
1419	004754	001404				BEQ	1\$; NO
1420	004756	005137	001114		3\$:	COM	AOB1	; ALTERNATE PORTS
1421	004762	000137	006014		2\$:	JMP	EXTPPR	; LOOP
1422	004766	000137	005004		1\$:	JMP	RANEX	; DO NEXT TEST
1423								
1424	004772	000137	004454		READCT:	JMP	ESH	; CONT. READING
1425	004776	052737	100000	001216	RANEL:	BIS	#BIT15,FLAG3	; SET LOOP IN RANDOM TEST GOT
1426								; HERE BY STARTING AT LOC 240

```

1427 005004
1428
1429
1430
1431
1432
1433
1434 005004 104400
1435 005006 032737 000004 001126
1436 005014 001402
1437 005016 000137 006014
1438 005022 052737 000400 001126
1439 005030 012737 020000 017360
1440 005036 013737 017360 001116
1441 005044 005737 001212
1442 005050 001402
1443 005052 005037 177572
1444 005056 012737 000042 001136
1445 005064 104426
1446 005066 012737 176030 001156
1447 005074 012737 005634 001152
1448 005102 004737 006722
1449 005106 012777 000340 173750
1450 005114 012737 005216 001010
1451 005122 012737 000001 001222
1452 005130 013701 017360
1453 005134 004537 007766
1454 005140 017737 012214 001134
1455 005146 042737 170000 001134
1456 005154 052737 000003 001126
1457 005162 004737 006510
1458 005166 013737 001130 001222
1459 005174 013701 017360
1460 005200 004537 007766
1461 005204 013737 017360 001140
1462 005212 005037 001200

```

```

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
;*****
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 #SR0 ;TURN IT OFF
      MOV #42,PATNU ;DO RANDOM COMPARE
      CLRDV ;INIT DRIVE
      MOV #-1000.,PASSC ;SET UP PASS COUNT
      MOV #RWRED,WRDER ;SET UP FOR HARD ERROR
      JSR PC,VECTR ;SETUP INTERRUPT VECTOR
      MOV #340,STATUS
      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,WHTHU ;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

```

K03

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND
CZRSCF.P11 16-DEC-77 15:10

RELIABILITY TEST
TST3 RANDOM ADDRESS RANDOM DATA TEST

MACY11 30(1046) 20-DEC-77 15:36 PAGE 37

SEQ 0036

1463	005216	104414			WRERR:	ERCLR			
1464	005220	012737	000161	001172		MOV	#161,CMD		;WRITE WITH I/E
1465	005226	104416				DKCMD			;WRITE
1466	005230	004737	011712			JSR	PC,WATT		;WAIT FOR INTERRUPT
1467	005234	032737	001000	001126	2\$:	BIT	#BIT9,FLAG		;WAS THERE AN ERROR?
1468	005242	001435				BEQ	WPRCK1		;NO
1469	005244	032777	000100	173554		BIT	#BIT6,JSWR		;TYPE RETRY ?
1470	005252	001003				BNE	5\$;YES
1471	005254	005737	001200			TST	LOPCNT		;FIRST TIME?
1472	005260	001013				BNE	6\$;NO
1473	005262	104430			5\$:	LOGW			;LOG WRITE ERROR
1474	005264	005237	001200			INC	LOPCNT		;SET ERROR FLAG
1475	005270	004737	014146			JSR	PC,PRNT		;TYPEOUT?
1476	005274	001004				BNE	3\$;YES
1477	005276	104402	000674			TYPE	,RANDM		
1478	005302	104402	000564			TYPE	,WRERR		
1479	005306	104044			3\$:	HLT	!DS!DA		
1480	005310	104400			6\$:	SCOPE			
1481	005312	005337	001126			DEC	FLAG		
1482	005316	032737	000003	001126		BIT	#3,FLAG		
1483	005324	001334				BNE	WRERR		;RETRY
1484	005326	004737	011402			JSR	PC,WTNO		;TYPE CAN NOT WRITE
1485	005332	000137	005736			JMP	EXRAX		;GET NEW NUMBER

L03

1486	005336	005737	001200		WRRCK1:	TST	LOPCNT		; ANY ERRORS?
1487	005342	001402				BEQ	1\$; NO
1488	005344	004737	011734			JSR	PC, TYPREC		; TYPE RECOVERED
1489	005350	104400			1\$:	SCOPE			
1490	005352	005037	001200			CLR	LOPCNT		; CLEAR LOOP COUNT
1491	005356	104414			WRRCK:	ERCLR			; CLEAR RS REG IF ERRORS
1492	005360	012737	000151	001172		MOV	#151, CMD		; WRITE CHECK WITH I/E
1493	005366	104416				DKCMD			; WRITE CHECK
1494	005370	004737	011712			JSR	PC, WATT		; WAIT FOR INTERRUPT
1495	005374	032737	001000	001126	4\$:	BIT	#BIT9, FLAG		; ERROR?
1496	005402	001453				BEQ	1\$; NO
1497	005404	032777	000100	173414		BIT	#BIT6, ASWR		; TYPE ALL RETRYS?
1498	005412	001003				BNE	2\$; YES
1499	005414	005737	001200			TST	LOPCNT		; FIRST ERROR?
1500	005420	001030				BNE	5\$; NO
1501	005422	104434			2\$:	LOGWC			; LOG WRITE CK
1502	005424	004737	014146			JSR	PC, PRNT		; TYPEOUT?
1503	005430	001052				BNE	6\$; NO
1504	005432	104402	000674			TYPE	, RANDM		
1505	005436	104402	000574			TYPE	, WCKERR		
1506	005442	017737	173372	001222		MOV	ASBBA, WORK		; GET CORRECT BA
1507	005450	162737	000002	001222		SUB	#2, WORK		
1508	005456	104402	005462			TYPE	, +2		; .ASCIZ <15><12>"(BA)=""
1509	005472	017746	173524			MOV	ASBBA, -(6)		; PUT ASBBA ON STACK
1510	005476	104404				TYPE0			; TYPE STACK IN OCTAL
1511	005500	104026				HLT	!DB!DA!BA		; BA=MEMORY LOC +2 OF ACTUAL WORD
1512	005502	005237	001200		5\$:	INC	LOPCNT		; INC RETRY COUNT
1513	005506	022737	000010	001200		CMP	#10, LOPCNT		; LAST ONE YET?
1514	005514	001320				BNE	WRRCK		; NO
1515	005516	104402	000522			TYPE	, UNRECO		
1516	005522	005037	001200			CLR	LOPCNT		; CLEAR LOPCNT
1517	005526	000137	005736			JMP	EXRAX		; GET NEW NUMBER
1518	005532	005737	001200		1\$:	TST	LOPCNT		; ANY ERRORS?
1519	005536	001407				BEQ	6\$; NO
1520	005540	104402	000616			TYPE	, RECOV		
1521	005544	013746	001200			MOV	LOPCNT, -(6)		; GET NUMBER
1522	005550	104406				TYPES			; TYPE IT
1523	005552	104402	000636			TYPE	, CRLFLF		
1524	005556	104400			6\$:	SCOPE			
1525	005560	052737	000003	001126		BIS	#3, FLAG		; SET COUNTER

M03

1526	005566	104400				SCOPE		
1527	005570	005037	001200			CLR	LOPCNT	; CLEAR COUNTER
1528	005574	004537	011230			JSR	RS,CLEAR	; CLEAR BUFFER
1529	005600	104414			RREAD:	ERCLR		; CLEAR RS REG IF ERRORS
1530	005602	012737	000171	001172		MOV	#171,CMD	; READ WITH I/E
1531	005610	104416				DKCMD		; READ
1532	005612	004737	011712			JSR	PC WATT	; WAIT FOR INTERRUPT
1533	005616	032777	010000	173202		BIT	#BIT12,ASWR	; COMPARE ?
1534	005624	001003				BNE	RWRED	; NO
1535	005626	004537	010374			JSR	RS,COMPARE	; YES
1536	005632	000400				BR	RWRED	; CONT
1537	005634	032737	001000	001126	RWRED:	BIT	#BIT9,FLAG	; IS THERE AN ERROR?
1538	005642	001435				BEQ	EXRAX	; NO
1539	005644	104432				LOGR		; LOG READ ERR
1540	005646	032777	000100	173152	1\$:	BIT	#BIT6,ASWR	; TYPE ALL ERRORS?
1541	005654	001016				BNE	2\$; YES
1542	005656	032777	001000	173142		BIT	#BIT9,ASWR	; LOOP ON ERROR?
1543	005664	001012				BNE	2\$; YES
1544	005666	005737	001200			TST	LOPCNT	; FIRST ERROR?
1545	005672	001010				BNE	3\$; NO
1546	005674	004737	014146			JSR	PC,PRNT	; TIMEOUT?
1547	005700	001004				BNE	2\$; NO
1548	005702	104402	000674			TYPE	,RANDM	
1549	005706	104402	000607			TYPE	,RDERR	
1550	005712	104006			2\$:	HLT	!DB!DA	
1551	005714	104400			3\$:	SCOPE		
1552	005716	005237	001200			INC	LOPCNT	; UPDATE COUNTER
1553	005722	022737	000010	001200		CMP	#10,LOPCNT	; LAST TRY YET?
1554	005730	001323				BNE	RREAD	; RETRY
1555	005732	004737	006446			JSR	PC,NOREC	; TYPE UNRECOVERABLE
1556	005736	005737	001200		EXRAX:	TST	LOPCNT	; ANY ERRORS?
1557	005742	001402				BEQ	EXRXX	; NO
1558	005744	004737	011734			JSR	PC,TYPREC	; TYPE RECOVERED
1559	005750	104400			EXRXX:	SCOPE		
1560	005752	005237	001156			INC	PASSC	; +1 PASS COUNT
1561	005756	001402				BEQ	1\$; IS TEST DONE?
1562	005760	000137	005114			JMP	WRLG1	; NO
1563	005764	005037	001136		1\$:	CLR	PATNU	; END OF TEST
1564	005770	042737	000400	001126		BIC	#BIT8,FLAG	; CLEAR TEST FLAG
1565	005776	032737	100000	001216		BIT	#BIT15,FLAG3	; LOOP ON THIS TEST?
1566	006004	001402				BEQ	+6	; NO
1567	006006	000137	005004			JMP	RANEX	; YES

1568
1569
1570
1571
1572
1573
1574
1575
1576
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

006012 104400
006010 005037 001134
006020 104426
006022 032737 004000 001126
006030 001404
006032 004737 002134
006036 000137 003170
006042 004737 002204

006046 032737 001000 001126
006054 001404
006056 104426
006060 042737 001000 001126
006066 000002

006070 013777 001134 172744
006076 005037 001176
006102 032737 020000 001122
006110 001021
006112 032737 000004 001126
006120 001415
006122 022737 000042 001136
006130 001411
006132 052777 000010 172674
006140 005077 172672
006144 012777 017360 172666
006152 000435
006154 013777 001140 172656
006162 013702 001130
006166 005402
006170 010277 172642
006174 032737 000400 001126
006202 001033
006204 005737 001212
006210 001416
006212 032737 000040 001122
006220 001403
006222 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 DMA
        CLRDV ;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 IS ;NO
        CLRDV ;CLEAR ALL ERRORS
        BIC #BIT9,FLAG ;CLEAR ERROR FLAG
IS: 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$: MOV BUF,DRSBA ;LOAD (CMA) BUSS ADDRESS
        MOV WRDCT,R2 ;GET NEGATIVE
        NEG R2 ;WORD COUNT
        MOV R2,DRSWC ;LOAD WC
        BIT #BIT8,FLAG ;RANDOM TEST?
        BNE IS ;YES A PORT ONLY WITH NO MEM MGMT
        TST MAVA ;MEM MGMT AVAILIABLE?
        BEQ 2$ ;NO
        BIT #BIT5,FLAG2 ;SET A17 IN RSCS1
        BEQ 3$ ;NO
        BIS #BIT9,CMD ;YES
```



```

1620 006230 032737 000020 001122 3$: BIT #BIT4,FLAG2 ;SET A16?
1621 006236 001403 BEQ 2$ ;NO
1622 006240 052737 000400 001172 BIS #BIT8,CMD ;YES
1623 006246 032737 000100 001126 2$: BIT #BIT6,FLAG ;MULTI PORT?
1624 006254 001406 BEQ 1$ ;NO
1625 006256 005737 001114 TST AOB1 ;TEST A OR B PORT?
1626 006262 001403 BEQ 1$ ;A PORT
1627 006264 052737 002000 001172 BIS #BIT10,CMD ;B PORT
1628 006272 013777 001172 1$: MOV CMD,ARSCS1 ;LOAD FUNCTION REG.
1629 006300 000002 RTI ;RETURN FROM TRAP
1630
1631 ;RH11 DISK INTERRUPT HANDLER
1632 ;ROUTINE CONTINUES ON ERRORS
1633
1634 006302 042737 001000 001126 DKINT: BIC #BIT9,FLAG ;CLEAR ERROR BIT
1635 006310 005777 172516 TST ARSCS1 ;TEST FOR ERROR
1636 006314 100401 BMI 2$
1637 006316 000425 BR INTEXT ;JUMP IF NO ERRORS
1638 006320 017702 172506 2$: MOV ARSCS1,R2 ;GET CONTENTS OF CS1
1639 006324 042702 037777 BIC #37777,R2 ;CLEAR ALL BUT SC AND TRE
1640 006330 022702 140000 CMP #140000,R2 ;IS SC AND TRE BOTH SET?
1641 006334 001413 BEQ TRUERR ;YES THERE IS SOME KIND OF XFER ERROR
1642 006336 032777 100000 172500 BIT #100000,ARSDS ;IS THE ATA BIT SET?
1643 006344 001007 BNE TRUERR ;YES
1644 006346 104140 HLT !AS!DS ;WRONG UNIT INTERRUPTED
1645 ;IF YOU HAVE JUST POWERED UP A DRIVE OR
1646 ;ARE RUNNING THE POWER FAIL TEST,
1647 ;INTERRUPTS WILL OCCUR FROM DRIVES OTHER
1648 ;THAN THE UNIT UNDER TEST. IF THIS TYPEOUT
1649 ;SHOWS NO ERRORS IN THE REGISTERS OF THE DRIVE
1650 ;UNDER TEST, THAT DRIVE IS OK
1651 006350 012777 177777 172472 1$: MOV #-1,ARSDS ;CLEAR ALL ATA BITS
1652 006356 013716 001152 MOV HDRR,(SP) ;GET RETURN ADD.
1653 006362 000002 RTI ;RETRY
1654 006364 052737 001000 001126 TRUERR: BIS #BIT9,FLAG ;SET ERORR BIT
1655 006372 032777 004000 172426 INTEXT: BIT #BIT11,ASWR ;HALT ON COMPLETION FLAG
1656 006400 001401 BEQ .+4
1657 006402 000000 HALT ;YES BIT 11 SET IN SWR HALT
1658 006404 032737 002000 001122 BIT #BIT10,FLAG2 ;WAIT IN BACKGROUND TEST?
1659 006412 001402 BEQ 1$ ;NO
1660 006414 012716 012360 2$: MOV #NPRRET,(SP) ;MODIFY RETURN ADD.
1661 006420 010637 001176 1$: MOV SP,INTFLG ;SET INT FLG
1662 006424 000002 RTI ;EXIT
1663
1664 ;ROUTINE TO SET UP TRACK # FROM OPTION
1665 ;ENTER FROM JSR RS, OPDSEL
1666
1667 006426 032737 000040 001126 OPDSEL: BIT #BIT5,FLAG ;OPTIONAL DMA?
1668 006434 001403 BEQ 1$ ;NO
1669 006436 013737 001142 001134 MOV TDMA,DMA ;GET OPT. DMA
1670 006444 000205 1$: RTS RS ;EXIT
    
```

```

1671
1672
1673          .EVEN
1674
1675 006446 004737 014146      NOREC: JSR      PC,PRNT      ;TYPEOUT?
1676 006452 001002                BNE      1$      ;NO
1677 006454 104402 000522                TYPE     ,UNRECO
1678 006460 005037 001200      1$: CLR      LOPCNT   ;CLEAR LOOP COUNTER
1679 006464 000207                RTS      PC
1680
1681 006466 052737 040000 001122  FNDRYP: BIS      #BIT14,FLAG2 ;SET CHECK DRIVE TYPE FLAG
1682 006474 004737 001746                JSR      PC,UNTYF ;CHECK DRIVE TYPE FLAG
1683 006500 042737 040000 001122                BIC      #BIT14,FLAG2 ;CLEAR DRIVE TYPE FLAG
1684 006506 000207                RTS      PC
1685 006510 022737 000004 001166  WHTHU: CMP      #4,RS04DT ;RS03LA?
1686 006516 001004                BNE      1$      ;NO
1687 006520 012737 000040 001130                MOV      #40,WRDCT ;GET WORD COUNT
1688 006526 000411                BR       2$
1689 006530 012737 000200 001130  1$: MOV      #200,WRDCT ;RS04
1690 006536 005737 001166                TST     RS04DT ;RS04?
1691 006542 001003                BNE      2$      ;YES
1692 006544 012737 000100 001130                MOV      #100,WRDCT ;NO
1693 006552 000207                RTS      PC
1694
1695

```

```

1696 ;ROUTINE TO CALCULATE VITURAL ADDR
1697
1698 006554 000302 PHYCOV: SWAB R2 ;CALCULATE FROM PHYSICAL ADDR
1699 006556 004737 011700 JSR PC,RRR2
1700 006562 006002 ROR R2
1701 006564 042702 177770 BIC #177770,R2 ;GET REG #
1702 006570 032777 000400 172234 BIT #BIT8,RSRCS1 ;IS A16 SET?
1703 006576 001402 BEQ 1$ ;NO
1704 006600 052702 000010 BIS #BIT3,R2 ;YES
1705 006604 032777 001000 172220 1$: BIT #BIT9,RSRCS1 ;IS A17 SET?
1706 006612 001402 BEQ 2$ ;NO
1707 006614 052702 000020 BIS #BIT4,R2 ;YES
1708 006620 013737 001070 001224 2$: MOV STAMEM,WORK1 ;GET BANK # FOR -A- PORT
1709 006626 005737 001114 TST AOB1 ;ARE WE ON -A- PORT?
1710 006632 001403 BEQ 3$ ;YES
1711 006634 013737 001074 001224 MOV STBCOM,WORK1 ;NO -B- PORT
1712 006642 163702 001224 3$: SUB WORK1,R2 ;GET STARTING BANK #
1713 006646 062702 000001 ADD #1,R2 ;GET OFFSET FOR REG #
1714 006652 000302 SWAB R2 ;GET BANK # INTO
1715 006654 006102 ROL R2 ;UPPER BITS
1716 006656 006102 ROL R2
1717 006660 006102 ROL R2
1718 006662 006102 ROL R2
1719 006664 006102 ROL R2
1720 006666 017737 172146 001224 MOV RSRBA,WORK1 ;GET OFFSET FOR ADDR IF ANY
1721 006674 162737 000002 001224 SUB #2,WORK1 ;CORRECT IT
1722 006702 042737 160000 001224 BIC #160000,WORK1 ;CLEAR JUNK
1723 006710 050237 001224 BIS R2,WORK1 ;GET REG NO
1724 006714 013702 001224 MOV WORK1,R2
1725 006720 000207 RTS PC
1726
1727 006722 012777 006302 172132 VECTR: MOV #DKINT,RSVVEC ;SETUP INTERRUPT VECTORS
1728 006730 013737 001066 177776 MOV PRIORITY,PS ;PRIORITY 4
1729 006736 000207 RTS PC
1730
1731 ;THIS ROUTINE IS USED FOR DELAYING THE START OF THIS PROGRAM
1732 ;IF POWER FAILED DURING TESTING. THIS WILL GIVE THE DRIVES TIME TO GET UP
1733 ;TO SPEED. THE DELAY WILL BE ABOUT 3-5 MINUTES DEPENDING UPON THE PROCESSOR
1734
1735 006740 012737 000677 001222 TIMUP: MOV #677,WORK
1736 006746 012737 177777 001224 1$: MOV #177777,WORK1
1737 006754 000240 2$: NOP
1738 006756 005337 001224 DEC WORK1
1739 006762 001374 BNE 2$
1740 006764 005337 001222 DEC WORK
1741 006770 001366 BNE 1$
1742 006772 000137 003154 JMP ADTST

```

E04

```

1743 ;ROUTINE TO SETUP DISK BUFFERS
1744 ;ADD WORD COUNT TO STARTING DISK ADDRESSES
1745 ;COMPARE CALCULATED ADDRESS TO TERMINATING ADDRESS
1746
1747 006776 032737 000040 001126 DISBUF: BIT #BITS,FLAG ;DID OPERATOR SELECT PATTERNS
1748 007004 001402 BEQ 2$ ;NO
1749 007006 000137 007270 JMP BUFEXIT ;YES
1750 007012 022737 000042 001136 2$: CMP #42,PATNU ;RANDOM PATTERN?
1751 007020 001443 BEQ 1$ ;YES
1752 007022 032737 000004 001126 BIT #BIT2,FLAG ;MAX TST?
1753 007030 001437 BEQ 1$ ;NO
1754 007032 022737 000004 001166 CMP #4,RS04DT ;RS03LA?
1755 007040 001010 BNE 4$ ;NO
1756 007042 022737 004000 001134 CMP #4000,DMA ;DONE YET?
1757 007050 001507 BEQ BUFEXIT ;YES
1758 007052 062737 004000 001134 ADD #4000,DMA ;UPDATE DMA
1759 007060 000207 RTS PC
1760 007062 005737 001166 4$: TST RS04DT ;RS04?
1761 007066 001010 BNE 3$ ;YES
1762 007070 022737 006000 001134 CMP #6000,DMA ;RS03
1763 007076 001474 BEQ BUFEXIT ;DONE GET OUT
1764 007100 062737 002000 001134 ADD #2000,DMA ;UPDATE DMA
1765 007106 000207 RTS PC ;EXIT
1766 007110 022737 007000 001134 3$: CMP #7000,DMA ;DONE YET?
1767 007116 001464 BEQ BUFEXIT ;YES
1768 007120 062737 001000 001134 ADD #1000,DMA ;UPDATE ADDR
1769 007126 000207 RTS PC
1770 007130 004737 007504 1$: JSR PC,BLSZ ;DEFINE BLOCK SIZE
1771 007134 013737 001154 001224 MOV BLOCK,WORK1
1772 007142 005237 001134 INCSEC: INC DMA ;+1 SECTOR COUNT
1773 007146 022737 010000 001134 CMP #10000,DMA ;DONE YET?
1774 007154 001445 BEQ BUFEXIT ;YES
1775 007156 005337 001154 DEC BLOCK ;-1 FROM BLOCK COUNT
1776 007162 001401 BEQ COMDAR ;CMP DMA TO RSDA
1777 007164 000766 BR INCSEC ;RECYCLE
1778 007166 032737 001000 001126 COMDAR: BIT #BIT9,FLAG ;ANY ERRORS?
1779 007174 001401 BEQ 1$ ;NO ERRORS DO COMPARE ON RSDA
1780 007176 000207 RTS PC ;ERRORS DO NOT COMPARE RSDA
1781 007200 023777 001134 171634 1$: CMP DMA,RSDA ;COMPARE RSDA WITH DMA
1782 007206 001425 BEQ CMDAE ;SHOULD BE EQUAL
1783 007210 104432 LOGR ;AFTER TRANSFER RSDA AND DMA SHOULD BE =
1784 ;IF NOT, RSDA IS NOT CORRECT. DMA CONTAINS
1785 ;WHAT RSDA SHOULD =
1786 007212 013701 001134 MOV DMA,GOOD ;GET DMA FOR CORRECT ANS IN GOOD
1787 007216 017700 171620 MOV RSDA,BAD ;GET RSDA INTO BAD
1788 007222 104000 HLT ;RSDA=BAD DMA=GOOD SEE COMMENTS 7 LINES ABOVE
1789 007224 004737 014146 JSR PC,PRNT ;TYPEOUT?
1790 007230 001014 BNE CMDAE ;NO
1791 007232 011637 001222 MOV (SP),WORK ;GET TEST PC FROM WHERE IT CAME
1792 007236 104402 007242 TYPE .+2 ;ASCIZ " TST PC="
1793 007254 013746 001222 MOV WORK,-(6) ;PUT WORK ON STACK
1794 007260 104406 TYPES ;TYPE STACK IN OCTAL - SUPRESS
1795 007262 105737 001126 CMDAE: TSTB FLAG ;LAST DISK BUFFER?
1796 007266 100032 BPL BUFINX ;NO
  
```

1797	007270	005037	001134		BUFEXIT: CLR	DMA	; CLEAR ADDRESS BITS LAST DISK BUFFER
1798	007274	062716	000002		ADD	#2, (6)	; INC STOCK POINTER
1799	007300	042737	000200	001126	AKH: BIC	#200, FLAG	; CLEAR LAST DISK BUFFER FLAG
1800	007306	032737	000400	001126	BIT	#BIT8, FLAG	; RANDOM TEST OR ADDR TEST?
1801	007314	001404			BEQ	1\$; NO
1802	007316	013737	001144	001130	2\$: MOV	SWRDCT, WRDCT	
1803	007324	000466			BR	EXTDR	; EXIT
1804	007326	032737	000100	001126	1\$: BIT	#BIT6, FLAG	; MULTI PORT?
1805	007334	001770			BEQ	2\$; NO
1806	007336	005737	001114		TST	AOB1	; A OR B PORT?
1807	007342	001765			BEQ	2\$; A PORT
1808	007344	013737	001112	001130	MOV	WDCTB, WRDCT	; B PORT
1809	007352	000453			BR	EXTDR	; GET OUT
1810	007354	005037	001226		BUF INX: CLR	WORK2	; CLEAR WORK2 FOR BLOCK COUNTER
1811	007360	013702	001134		MOV	DMA, R2	; PUT WORKING DISK ADD INTO WORK
1812	007364	005237	001226		XINCSEC: INC	WORK2	; INCREMENT BLOCK COUNT
1813	007370	022702	007777		CMP	#7777, R2	; CMP FOR LAST SECTOR
1814	007374	001405			BEQ	XINCSUR	; +1 SURFACE LAST SECTOR BRANCH
1815	007376	005202			INC	R2	; INC DMA
1816	007400	005337	001224		DEC	WORK1	; DEC BLOCK COUNT
1817	007404	001367			BNE	XINCSEC	; FILLED STANDARD BUFFER YET?
1818	007406	000734			BR	AKH	; WILL TAKE STANDARD SIZE WORD COUNT
1819	007410	013737	001226	001130	XINCSUR: MOV	WORK2, WRDCT	; SETTING UP BLOCK COUNT
1820	007416	000241			CLC		; FOR NON STANDARD BUFFER SIZE
1821	007420	006137	001130		ROL	WRDCT	
1822	007424	006137	001130		ROL	WRDCT	
1823	007430	006137	001130		ROL	WRDCT	
1824	007434	006137	001130		ROL	WRDCT	
1825	007440	006137	001130		ROL	WRDCT	
1826	007444	022737	000004	001166	CMP	#4, RS04DT	; RS03LA?
1827	007452	001410			BEQ	1\$; YES
1828	007454	000241			CLC		
1829	007456	006137	001130		ROL	WRDCT	
1830	007462	005737	001166		TST	RS04DT	; RS04?
1831	007466	001402			BEQ	1\$; NO
1832	007470	006137	001130		ROL	WRDCT	; YES
1833	007474	052737	000200	001126	1\$: BIS	#200, FLAG	; SET LAST DISK BUFFER FLAG
1834	007502	000207			EXTDR: RTS	PC	; EXIT

```

1835 ;THIS ROUTINE CONVERTS A WORD COUNT TO A BLOCK COUNT
1836 007504 022737 000004 001166 BLSZ:  CMP    #4,RS04DT    ;RS03LA
1837 007512 001004                BNE    3$                ;NO
1838 007514 012737 000037 001154        MOV    #37,BLOCK        ;YES
1839 007522 000411                BR     2$                ;CONTINUE
1840 007524 012737 000177 001154 3$:   MOV    #177,BLOCK        ;SETUP FOR RS04
1841 007532 005737 001166                TST    RS04DT           ;RS04?
1842 007536 001003                BNE    2$                ;YES
1843 007540 012737 000077 001154 1$:   MOV    #77,BLOCK        ;PUT SECTOR SIZE INTO BLOCK
1844 007546 013702 001130 2$:   MOV    WRDCT,R2         ;FETCH WORD COUNT
1845 007552 033702 001154                BIT    BLOCK,R2         ;ARE THEY EQUAL?
1846 007556 001406                BEQ    RORBLK           ;YES
1847 007560 043702 001154                BIC    BLOCK,R2         ;SET UP BLOCK OVERFLOW
1848 007564 005237 001154                INC    BLOCK
1849 007570 063702 001154                ADD    BLOCK,R2
1850 007574 000241                RORBLK: CLC
1851 007576 006002                ROR    R2
1852 007600 022737 000004 001166        CMP    #4,RS04DT        ;RS03LA
1853 007606 001003                BNE    2$                ;NO
1854 007610 004737 011700                JSR    PC,RRR2          ;YES
1855 007614 000410                BR     1$                ;CONTINUE
1856 007616 000241                2$:   CLC
1857 007620 006002                ROR    R2
1858 007622 004737 011700                JSR    PC,RRR2
1859 007626 005737 001166                TST    RS04DT           ;RS04?
1860 007632 001401                BEQ    1$                ;NO
1861 007634 006002                ROR    R2                ;YES
1862 007636 010237 001154 1$:   MOV    R2,BLOCK        ;BLOCK COUNT
1863 007642 000207                RTS    PC                ;EXIT
1864
1865 ;ROUTINE TO SELECT DATA PATTERNS FOR TEST
1866 ;ENTER FROM JSR R5,PASEL
1867 007644 012737 010300 000004 PASEL: MOV    #MEM,#4          ;SETUP TRAP
1868 007652 012737 000340 000006        MOV    #340,#6         ;VECTOR
1869 007660 013700 001136        MOV    PATNU,R0        ;SET UP PATTERN NUMBER
1870 007664 010003                MOV    R0,R3           ;GET PATTERN #
1871 007666 000241                CLC                    ;MAKE IT =
1872 007670 006003                ROR    R3              ;TO PATTERN # IN LISTING
1873 007672 010377 171132        MOV    R3,@DISPLAY    ;DISPLAY PATTERN #
1874 007676 013737 001130 001222        MOV    WRDCT,WORK     ;SET UP WORK
1875 007704 013701 001116        MOV    VADDR,R1       ;LOC. OF OUTBUFFER
1876 007710 022700 000042 1$:   CMP    #42,R0         ;TEST FOR RANDOM DATA NUMBER
1877 007714 001424                BEQ    RANDOM          ;GO GENERATE RANDOM DATA
1878 007716 032737 000004 001126        BIT    #BIT2,FLAG     ;MAX TST?
1879 007724 001404                BEQ    2$              ;NO
1880 007726 016037 000254 017360        MOV    PATD(0),OUTBUF ;GET PATTERN
1881 007734 000205                RTS    R5
1882 007736 016000 000254 2$:   MOV    PATD(0),R0
1883 007742 010021 FILDAT: MOV    R0,(1)+        ;FILL BUFFER
1884 007744 005337 001222                DEC    WORK            ;DEC. WORK COUNT
1885 007750 001374                BNE    FILDAT         ;LOAD NEXT WORD
1886 007752 012737 000006 000004 PASEX: MOV    #6,#4          ;RESTORE
1887 007760 005037 000006                CLR    #6              ;TRAP
1888 007764 000205                RTS    R5              ;BUFFER FULL

```

;RANDOM DATA GENERATOR SUBROUTINE

1889											
1890											
1891	007766	013737	010132	010136	RANDOM:	MOV	LONUM, LOSAV				
1892	007774	013737	010134	010140		MOV	HINUM, HISAV				
1893	010002	013700	010132		RAND1:	MOV	LONUM, R0		;SET UP R0 WITH 5 DIGITS LOW		
1894	010006	013704	010134			MOV	HINUM, R4		;SET UP R1 WITH 5 DIGITS HIGH		
1895	010012	012703	000007			MOV	#7, R3		;SET UP SHIFT COUNT		
1896	010016	005002				CLR	R2		;CLEAR R2		
1897	010020	006300			SHIFT:	ASL	R0		;SHIFT R0 LEFT AND		
1898	010022	006104				ROL	R4		;ROTATE CARRY INTO LSB OF R1 INTO		
1899	010024	006102				ROL	R2		;ROTATE CARRY OUT OF R1 INTO R2		
1900	010026	005303				DEC	R3		;DECREMENT R3		
1901	010030	001373				BNE	SHIFT		;CONTINUE SHIFT LOOP		
1902	010032	063700	010132			ADD	LONUM, R0		;ADDN IN NUMBER TO MAKE X 129		
1903	010036	005504				ADC	R4		;PROPOGATE CARRY		
1904	010040	063704	010134			ADD	HINUM, R4		;ADDN IN NUMBER TO MAKE X 129		
1905	010044	005502				ADC	R2		;PROPOGATE CARRY		
1906	010046	062700	001057			ADD	#1057, R0		;ADDN LOW CONSTANT		
1907	010052	005504				ADC	R4		;PROPOGATE CARRIES		
1908	010054	005502				ADC	R2		;PROPOGATE AGAIN		
1909	010056	062704	047401			ADD	#47401, R4		;ADDN HIGH CONSTANT		
1910	010062	005502				ADC	R2		;PROPOGATE CARRY		
1911	010064	062702	000006			ADD	#6, R2		;ADDN HIGHEST CONSTANT		
1912	010070	062700	000002			ADD	#2, R0		;REPRIME R0 WITH HIGH DIGIT		
1913	010074	005504				ADC	R4		;PROPOGATE CARRY		
1914	010076	010037	010132			MOV	R0, LONUM		;PUT R0 BACK IN LONUM		
1915	010102	010021				MOV	R0, (1)+		;HOLD LONUM FOR PROGRAM		
1916	010104	005337	001222			DEC	WORK				
1917	010110	001406				BEQ	EXGEN				
1918	010112	010437	010134			MOV	R4, HINUM		;PUT R1 BACK IN HINUM		
1919	010116	010421				MOV	R4, (1)+		;HOLD HINUM FOR PROGRAM		
1920	010120	005337	001222			DEC	WORK				
1921	010124	001326				BNE	RAND1				
1922	010126	000137	007752			JMP	PASEX		;RETURN TO PROGRAM		
1923	010132	000000			EXGEN:	LONUM:	0				
1924	010134	000000				HINUM:	0				
1925	010136	000000				LOSAY:	0				
1926	010140	000000				HISAV:	0				
1927											
1928	010142	013737	001214	001144	RESTOR:	MOV	SAVWC, SWRDCT		;RESTORE ORIGINAL		
1929	010150	013737	001144	001130		MOV	SWRDCT, WRDCT		;WORD COUNT		
1930	010156	013737	001220	001112		MOV	SAVWCB, WDCTB				
1931	010164	000205				RTS	R5				

```

1932
1933
1934
1935 010166 013700 010274
1936 010172 013704 010276
1937 010176 012703 000007
1938 010202 005002
1939 010204 006300
1940 010206 006104
1941 010210 006102
1942 010212 005303
1943 010214 001373
1944 010216 063700 010274
1945 010222 005504
1946 010224 063704 010276
1947 010230 005502
1948 010232 062700 001057
1949 010236 005504
1950 010240 005502
1951 010242 062704 047401
1952 010246 005502
1953 010250 062702 000006
1954 010254 062700 000002
1955 010260 005504
1956 010262 010037 010274
1957 010266 010437 010276
1958 010272 000205
1959 010274 000000
1960 010276 000000
1961
1962
1963
1964 010300
1965 010300 104402 010304
1966 010320 005737 001114
1967 010324 001004
1968 010326 104402 010332
1969 010334 000403
1970 010336
1971 010336 104402 010342
1972 010344 012737 000006 000004
1973 010352 005037 000006
1974 010356 032777 100000 170442
1975 010364 001401
1976 010366 000000
1977 010370 000137 001234

```

```

;RANDOM DATA GENERATOR SUBROUTINE
;WHEN SWITCH = 0 WE COME HERE

```

```

RAND:  MOV LONUM1,R0      ;SET UP R0 WITH 5 DIGITS LOW
        MOV HINUM1,R4    ;SET UP R1 WITH 5 DIGITS HIGH
        MOV #7,R3        ;SET UP SHIFT COUNT
        CLR R2           ;CLEAR R2
SHIFT1: ASL R0           ;SHIFT R0 LEFT AND
        ROL R4           ;ROTATE CARRY INTO LSB OF R1 INTO
        ROL R2           ;ROTATE CARRY OUT OF R1 INTO R2
        DEC R3           ;DECREMENT R3
        BNE SHIFT1      ;CONTINUE SHIFT LOOP
        ADD LONUM1,R0    ;ADDN IN NUMBER TO MAKE X 129
        ADC R4           ;PROPOGATE CARRY
        ADD HINUM1,R4    ;ADDN IN NUMBER TO MAKE X 129
        ADC R2           ;PROPOGATE CARRY
        ADD #1057,R0     ;ADDN LOW CONSTANT
        ADC R4           ;PROPOGATE CARRIES
        ADC R2           ;PROPOGATE AGAIN
        ADD #47401,R4    ;ADDN HIGH CONSTANT
        ADC R2           ;PROPOGATE CARRY
        ADD #6,R2        ;ADDN HIGHEST CONSTANT
        ADD #2,R0        ;REPRIME R0 WITH HIGH DIGIT
        ADC R4           ;PROPOGATE CARRY
        MOV R0,LONUM1    ;PUT R0 BACK IN LONUM
        MOV R4,HINUM1    ;PUT R1 BACK IN HINUM
EXGEN1: RTS              ;RETURN TO PROGRAM
LONUM1: 0
HINUM1: 0

```

```

;TRAP OUT ROUTINE WHEN CREATING DATA BUFFER

```

```

MEM:    TYPE      .+2      ;.ASCIZ <15><12>"NXM PORT "
        TST      AOB1     ;FIND WHAT DATA BUFFER
        BNE      3$      ;BRANCH IF B
        TYPE      .+2      ;.ASCIZ "A"
        BR       4$
3$:     TYPE      .+2
4$:     MOV      #6,@#4    ;.ASCIZ "B"
        CLR      @#6      ;RESTORE
        BIT      #BIT15,@SWR ;TRAP
        BEQ      2$      ;HALT?
        HALT
2$:     JMP      @#BEGIN   ;NO

```



```

1978 ; THIS ROUTINE COMPARES THE DATA READ AGAINST THE DATA EXPECTED.
1979 ; ALL ERRORS ARE REPORTED TO THE OPERATOR. IF BIT 4 OF THE SWITCH
1980 ; REGISTER IS SET, THIS ROUTINE WILL CONTINUE COMPARING AFTER AN ERROR HAS BEEN
1981 ; FOUND AND WILL REPORT UP TO 8 VERIFY ERRORS WITHIN THE SAME INPUT OPERATION.
1982
1983 010374 012737 177770 001146 COMPAR: MOV #10,ERCOUNT ; ERROR RETRY COUNTER
1984 010402 052737 000010 001122 BIS #BIT3,FLAG2 ; DOING COMPARE
1985 010410 013737 001130 001226 MOV WRDCT,WORK2 ; GET THE WORD COUNT
1986 010416 013737 001116 001150 MOV VADDR,SAVE ; SET UP OUTBUFFER POINTER
1987 010424 005037 001174 CLR SWITCH ; CLEAR RANDOM PATTERN FLAG
1988 010430 013737 010136 010274 MOV LOSAV,LONUM1 ; GET RANDOM BASE NOS.
1989 010436 013737 010140 010276 MOV HISAV,HINUM1
1990 010444 005737 001136 TST PATNU ; TEST FOR PATTERN 0
1991 010450 001015 BNE 1$ ; NO
1992 010452 005037 001222 CLR WORK ; CLEAR COUNTER
1993 010456 062737 000001 001222 2$: ADD #1,WORK ; INC COUNTER
1994 010464 001003 BNE 3$ ; INTERRUPT YET?
1995 010466 104054 HLT !DA!WC!DS ; TIMED OUT NOT INTERRUPT
1996 010470 000137 001234 JMP @#BEGIN
1997 010474 005737 001176 3$: TST INTFLG ; TEST FOR INT
1998 010500 001766 BEQ 2$ ; WAIT FOR INT BEFORE COMPARING
1999 010502 000426 BR ; CONT
2000 010504 022737 000042 001136 1$: CMP #42,PATNU ; IS THIS RANDOM PATTERN?
2001 010512 001022 BNE CMPLP1 ; BRANCH IF YES
2002 010514 005737 001176 CMPLP: TST INTFLG ; INTERRUPT YET?
2003 010520 001775 BEQ CMPLP ; NO WAIT
2004 010522 005737 001174 TST SWITCH
2005 010526 001007 BNE 2$
2006 010530 004537 010166 JSR R5,RAND ; GET EVEN RANDOM WORD
2007 010534 013701 010274 MOV LONUM1,GOOD ; SET RANDOM PATTERN FLAG
2008 010540 010637 001174 MOV SP,SWITCH
2009 010544 000411 BR WRDCMP
2010 010546 005037 001174 2$: CLR SWITCH
2011 010552 013701 010276 MOV HINUM1,GOOD
2012 010556 000404 BR WRDCMP
2013 010560 013700 001136 CMPLP1: MOV PATNU,RO
2014 010564 016001 000254 MOV PATD(RO),GOOD
2015 010570 160177 170354 WRDCMP: SUB GOOD,@SAVE ; COMPARE DATA
2016 010574 001017 BNE WDERR ; WORD IN ERROR
2017 010576 005337 001226 WRDINC: DEC WORK2 ; DECREMENT THE WORD COUNT
2018 010602 001410 BEQ ADAM ; EXIT ROUTINE IF ZERO
2019 010604 062737 000002 001150 ADD #2,SAVE ; UPDATE PATTERN ADDRESS
2020 010612 022737 000042 001136 CMP #42,PATNU ; IS THIS RANDOM PATTERN
2021 010620 001735 BEQ CMPLP ; BRANCH IF YES
2022 010622 000762 BR WRDCMP ; COMPARE NEXT WORD
2023 010624 042737 000010 001122 ADAM: BIC #BIT3,FLAG2 ; DONE WITH COMPARE
2024 010632 000205 RTS R5 ; EXIT THIS ROUTINE

```

2025	010634	005737	001176		WDERR:	TST	INTFLG	; DID INTERRUPT OCCUR YET?
2026	010640	001753				BEQ	WRDCMP	; BRANCH IF NO
2027	010642	032777	000100	170156		BIT	#BIT6, ASWR	; TRY ALL?
2028	010650	001006				BNE	10\$; YES
2029	010652	005737	001200			TST	LOPCNT	; FIRST READ ERROR?
2030	010656	001403				BEQ	10\$; YES
2031	010660	005777	170146			TST	ASCS1	; ANY ERRORS?
2032	010664	100757				BMI	ADAM	; YES DO NOT COMPARE
2033	010666	060177	170256		10\$:	ADD	GOOD, ASAVE	
2034	010672	017700	170252			MOV	ASAVE, BAD	; GET GOOD DATA
2035	010676	104436				LOGC		; LOG COMPARE ERROR
2036	010700	032777	001000	170120		BIT	#BIT9, ASWR	; LOOP ON ERROR?
2037	010706	001401				BEQ	11\$; NO
2038	010710	005726				TST	(6)+	; YES UPDATE SP
2039	010712	004737	014146		11\$:	JSR	PC, PRNT	; TYPEOUT?
2040	010716	001007				BNE	3\$; NO
2041	010720	104402	010724			TYPE	,.+2	; .ASCIZ <15><12>"CMP ERR"
2042	010736	104000			3\$:	HLT		; DATA COMPARE ERROR
2043	010740	004737	014146			JSR	PC, PRNT	; HAD TO DO IT THIS WAY SO
2044	010744	001022				BNE	13\$; PROGRAM COULD LOOP ON ERROR
2045	010746	104402	010752			TYPE	,.+2	; .ASCIZ " ADDR="
2046	010762	005737	001212			TST	MMAVA	; IS MEM MGMT ON?
2047	010766	001406				BEQ	12\$; NO
2048	010770	013746	177776			MOV	PS, -(6)	; GET PS
2049	010774	013746	001150			MOV	SAVE, -(6)	; GET VIRTUAL ADDR
2050	011000	104412				TYPEA		; CONVERT TO PHY AND TYPE
2051	011002	000403				BR	13\$; CONT
2052	011004	013746	001150		12\$:	MOV	SAVE, -(6)	; GET ADDR
2053	011010	104406				TYPES		; TYPE IT
2054	011012	005037	001154		13\$:	CLR	BLOCK	; CLEAR THE BLOCK COUNTER
2055	011016	013702	001130			MOV	WRDCT, R2	; GET THE WORD COUNT
2056	011022	005202				INC	R2	; CORRECT FOR DA CALCULATIONS
2057	011024	163702	001226			SUB	WORK2, R2	; DETERMINE DISTANCE OF FAILURE INTO BUFFER
2058	011030	022737	000004	001166	2\$:	CMP	#4, RS04DT	; RS03LA?
2059	011036	001003				BNE	14\$; NO
2060	011040	162702	000040			SUB	#40, R2	
2061	011044	000410				BR	9\$; CONTINUE
2062	011046	005737	001166		14\$:	TST	RS04DT	; RS04?
2063	011052	001403				BEQ	7\$; NO
2064	011054	162702	000200			SUB	#200, R2	; RS03
2065	011060	000402				BR	9\$; CONT
2066	011062	162702	000100		7\$:	SUB	#100, R2	
2067	011066	100403			9\$:	BMI	8\$	
2068	011070	005237	001154			INC	BLOCK	; UPDATE BLOCK COUNT FOR EACH 400 WORDS
2069	011074	000755				BR	2\$	

2070	011076	022737	000004	001166	8\$:	CMP	#4,RS04DT	;RS04LA
2071	011104	001003				BNE	15\$;NO
2072	011106	062702	000040			ADD	#40,R2	;RESTORE POSITIVE #
2073	011112	000410				BR	6\$;CONTINUE
2074	011114	005737	001166		15\$:	TST	RS04DT	;RS04?
2075	011120	001403				BEQ	4\$;NO
2076	011122	062702	000200			ADD	#200,R2	;RS04
2077	011126	000402				BR	6\$;CONT
2078	011130	062702	000100		4\$:	ADD	#100,R2	;RESTORE POSITIVE NUMBER
2079	011134	013737	001134	001224	6\$:	MOV	DMA,WORK1	;GET HEAD AND SECTOR ADDRESS
2080	011142	063737	001154	001224	5\$:	ADD	BLOCK,WORK1	
2081	011150	004737	014146			JSR	PC,PRNT	;TYPEOUT?
2082	011154	001014				BNE	1\$;NO
2083	011156	104402	011162			TYPE	.+2	;ASCIZ " DA="
2084	011170	013746	001224			MOV	WORK1,-(6)	;PUT WORK1 ON STACK
2085	011174	104406				TYPES		;TYPE STACK IN OCTAL - SUPRESS
2086	011176	104402	011202			TYPE	.+2	;ASCIZ <15><12>
2087	011206	032777	000020	167612	1\$:	BIT	#BIT4,ASWR	;RETRY?
2088	011214	001405				BEQ	CLEAR	;NO
2089	011216	005237	001146			INC	ERCOUNT	;UPDATE ERROR COUNTER
2090	011222	001402				BEQ	CLEAR	
2091	011224	000137	010576			JMP	WRDINC	
2092	011230	032737	000004	001126	CLEAR:	BIT	#BIT2,FLAG	;XFER TEST?
2093	011236	001404				BEQ	3\$;NO
2094	011240	032737	010000	001126		BIT	#BIT12,FLAG	;READ?
2095	011246	001412				BEQ	2\$;NO
2096	011250	013700	001116		3\$:	MOV	VADDR,R0	;GET STARTING ADDR OF BUFFER
2097	011254	013701	001130			MOV	WRDCT,R1	;NOW
2098	011260	005020			1\$:	CLR	(R0)+	;CLEAR BUFFER
2099	011262	005301				DEC	R1	;COUNT LOCATIONS
2100	011264	001375				BNE	1\$;WAIT TILL DONE
2101	011266	042737	000010	001122		BIC	#BIT3,FLAG2	;DONE WITH COMPARE
2102	011274	000205			2\$:	RTS	R5	;NOW GET OUT
2103								
2104	011276	013737	001072	017360	APOINT:	MOV	SAVAST,OUTBUF	;SET STARTING ADDR FOR OUTBUF
2105	011304	013737	001072	001116		MOV	SAVAST,VADDR	;SAVE OUTBUF ADDR
2106	011312	005737	001212			TST	MMAVA	;MEM MGMT?
2107	011316	001411				BEQ	EXTT	;NO
2108	011320	013702	001100			MOV	SAVMGA,R2	;SET UP MEM MGMT
2109	011324	004737	011762		MMSET:	JSR	PC,STM2	;SETUP MEM MGMT
2110	011330	010237	001116			MOV	R2,VADDR	
2111	011334	013737	001120	017360		MOV	PHADDR,OUTBUF	
2112	011342	000207			EXTT:	RTS	PC	
2113								
2114	011344	013737	001112	001130	BPOINT:	MOV	WDCTB,WRDCT	;GET WC FOR B PORT
2115	011352	013737	001076	017360		MOV	SAVCPU,OUTBUF	
2116	011360	013737	001076	001116		MOV	SAVCPU,VADDR	
2117	011366	005737	001212			TST	MMAVA	;MEM MGMT AVAILABLE?
2118	011372	001763				BEQ	EXTT	;NO
2119	011374	013702	001104			MOV	SAVMGC,R2	
2120	011400	000751				BR	MMSET	

;TYPE CAN NOT WRITE BLOCK

2121
2122
2123 011402 004737 014146
2124 011406 001001
2125 011410 000000
2126 011412 005037 001200
2127 011416 000207

WTNO: JSR PC,PRNT ;TYPEOUT?
BNE 1\$;NO
HALT ;HALT CANT WRITE BLOCK
1\$: CLR LOPCNT ;CLEAR ERR COUNTER
RTS PC

;ROUTINE TO SET UP STARTING ADDRESS FOR ALL PORTS
;AND TO CREATE WORD COUNT MAX= 20K

2128
2129
2130
2131
2132 011420 013702 001070
2133 011424 005702
2134 011426 001001
2135 011430 005202
2136 011432 005737 001212
2137 011436 001021
2138 011440 000241
2139 011442 004737 011700
2140 011446 010237 001072
2141 011452 032737 000100 001126
2142 011460 001430
2143 011462 013702 001074
2144 011466 000241
2145 011470 004737 011700
2146 011474 010237 001076
2147 011500 000420
2148 011502 000302
2149 011504 006002
2150 011506 010237 001100
2151 011512 032737 000100 001126
2152 011520 001410
2153 011522 013702 001074
2154 011526 000302
2155 011530 006002
2156 011532 010237 001102
2157 011536 010237 001104
2158 011542 013702 001106
2159 011546 005202
2160 011550 013703 001070
2161 011554 160302
2162 011556 022702 000007
2163 011562 101411

EXTMEM: MOV STAMEM,R2 ;GET BANK #
TST R2 ;DID HE TYPE 0?
BNE 3\$;NO
INC R2 ;YES MAKE 1
3\$: TST MMAVA ;BRANCH IF MEM MGMT AVAILABLE
BNE 1\$
CLC
JSR PC,RRR2
MOV R2,SAVAST ;SAVE A STARTING ADDR
BIT #BIT6,FLAG ;IS THERA A B PORT?
BEQ 2\$;NO
MOV STBCOM,R2 ;YES GET STARTING
CLC
JSR PC,RRR2
MOV R2,SAVCPU ;SAVE IT
BR 2\$;GET WC
1\$: SWAB R2
ROR R2
MOV R2,SAVMGA ;SAVE ADDR FOR A PORT
BIT #BIT6,FLAG ;IS THERE B PORT?
BEQ 2\$;NO
MOV STBCOM,R2
SWAB R2
ROR R2
MOV R2,SAVMGB ;SAVE B STARTING ADDR
MOV R2,SAVMGC ;SAVE CPU STARTING ADDR
2\$: MOV SIZEAP,R2 ;GET 4K BLOCK COUNT
INC R2
MOV STAMEM, R3
SUB R3, R2
CMP #7,R2 ;IS IT GREATER THEN 20K?
BLOS 4\$;YES MAKE IT 20K

2164	011564	000241			8\$:	CLC		
2165	011566	006002				ROR	R2	;NO CONVERT TO WC
2166	011570	004737	011700			JSR	PC,RRR2	
2167	011574	042702	000077			BIC	#77,R2	;CLEAR BLOCK COUNT
2168	011600	010237	001144			MOV	R2,SWRDCT	;SAVE -A- PORT WC
2169	011604	000403				BR	5\$;CONT
2170	011606	012737	060000	001144	4\$:	MOV	#6000,SWRDCT	;MAKE 20K
2171	011614	032737	000100	001126	5\$:	BIT	#BIT6,FLAG	;IS THERE B PORT?
2172	011622	001425				BEQ	7\$;NO GET OUT
2173	011624	013702	001110			MOV	SIZEBP,R2	;GET B 4K COUNT
2174	011630	005202				INC	R2	
2175	011632	013703	001074			MOV	STBCOM, R3	
2176	011636	160302				SUB	R3, R2	
2177	011640	022702	000007			CMP	#7,R2	;IS IT GREATER THEN 20K?
2178	011644	101411				BLOS	6\$;YES MAKE 20K
2179	011646	000241			9\$:	CLC		
2180	011650	006002				ROR	R2	;NO CONVERT TO WC
2181	011652	004737	011700			JSR	PC,RRR2	
2182	011656	042702	000077			BIC	#77,R2	;CLEAR SECTOR COUNT
2183	011662	010237	001112			MOV	R2,WDCTB	;SAVE WC FOR -B- PORT
2184	011666	000403				BR	7\$;GET OUT
2185	011670	012737	060000	001112	6\$:	MOV	#60000,WDCTB	;MADE 20K WD
2186	011676	000207			7\$:	RTS	PC	
2187								
2188	011700	006002			RRR2:	ROR	R2	
2189	011702	006002				ROR	R2	
2190	011704	006002				ROR	R2	
2191	011706	006002				ROR	R2	
2192	011710	000207				RTS	PC	
2193								
2194	011712	032777	000200	167106	WATT:	BIT	#BIT7,@SWR	;WAIT IN BACKGROUND?
2195	011720	001003				BNE	1\$;NO
2196	011722	004737	012254			JSR	PC,XWAIT	;YES
2197	011726	000401				BR	2\$;CONT
2198	011730	000001			1\$:	WAIT		
2199	011732	000207			2\$:	RTS	PC	
2200								
2201	011734	004737	014146		TYPREC:	JSR	PC,PRNT	;TYPEOUT?
2202	011740	001007				BNE	1\$;NO
2203	011742	104402	000616			TYPE	RECOV	
2204	011746	013746	001200			MOV	(OPCNT,-(6)	;GET COUNT
2205	011752	104406				TYPES		;TYPE IT
2206	011754	104402	000636			TYPE	CRLF	
2207	011760	000207			1\$:	RTS	PC	

```

2208
2209 011762 005737 001212 STMM2: TST MMAVA ;MEM MGMT?
2210 011766 001002 BNE 3$ ;YES
2211 011770 000137 012246 JMP MDON ;GET OUT
2212 011774 005037 172340 3$: CLR @#KIPAR0
2213 012000 010237 001150 MOV R2,SAVE ;SAVE R2
2214 012004 010237 172342 MOV R2,@#KIPAR1
2215 012010 006302 ASL R2 ;CALCULATE PHYSICAL ADDR
2216 012012 006302 ASL R2
2217 012014 006302 ASL R2
2218 012016 006302 ASL R2
2219 012020 006302 ASL R2
2220 012022 042737 000040 001122 BIC #BITS,FLAG2 ;THIS BIT IS A17
2221 012030 103003 1$ BCC 1$ ;CLEAR A17?
2222 012032 052737 000040 001122 BIS #BITS,FLAG2 ;SET A17
2223 012040 042737 000020 001122 1$: BIC #BIT4,FLAG2 ;SET BIT 5 FOR A17
2224 012046 006302 ASL R2 ;CLEAR A16 FLAG
2225 012050 103003 BCC 2$ ;GET A16 BIT
2226 012052 052737 000020 001122 BIS #BIT4,FLAG2 ;CLEAR A16
2227 012060 010237 001120 2$: MOV R2,PHADDR ;SET FLAG FOR A16
2228 012064 013702 001150 MOV SAVE,R2 ;GET PHYSICAL ADDR
2229 012070 062702 000200 ADD #200,R2 ;SET UP MEM MGMT
2230 012074 010237 172344 MOV R2,@#KIPAR2
2231 012100 062702 000200 ADD #200,R2
2232 012104 010237 172346 MOV R2,@#KIPAR3
2233 012110 062702 000200 ADD #200,R2
2234 012114 010237 172350 MOV R2,@#KIPAR4
2235 012120 062702 000200 ADD #200,R2
2236 012124 010237 172352 MOV R2,@#KIPAR5
2237 012130 062702 000200 ADD #200,R2
2238 012134 010237 172354 MOV R2,@#KIPAR6
2239 012140 012737 077406 172300 MOV #200*256.-400+UP+RW,@#KIPDR0 ;SET KIPDR0=RW UP 200 BLOCKS
2240 012146 012737 077406 172302 MOV #200*256.-400+UP+RW,@#KIPDR1 ;SET KIPDR1=RW UP 200 BLOCKS
2241 012154 012737 077406 172304 MOV #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
2242 012162 012737 077406 172306 MOV #200*256.-400+UP+RW,@#KIPDR3 ;SET KIPDR3=RW UP 200 BLOCKS
2243 012170 012737 077406 172310 MOV #200*256.-400+UP+RW,@#KIPDR4 ;SET KIPDR4=RW UP 200 BLOCKS
2244 012176 012737 077406 172312 MOV #200*256.-400+UP+RW,@#KIPDR5 ;SET KIPDR5=RW UP 200 BLOCKS
2245 012204 012737 077406 172314 MOV #200*256.-400+UP+RW,@#KIPDR6 ;SET KIPDR6=RW UP 200 BLOCKS
2246 012212 012737 077406 172316 MOV #200*256.-400+UP+RW,@#KIPDR7 ;SET KIPDR7=RW UP 200 BLOCKS
2247 012220 012737 007600 172356 MOV #7600,@#KIPAR7
2248 012226 012702 020000 MOV #20000,R2
2249 012232 012737 012250 000250 MOV #MMABTO,@#MMVEC
2250 012240 012737 000001 177572 MDON: MOV #1,@#SR0 ;TURN ON MEM MGMT
2251 012246 000207 RTS PC
2252
2253 ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
2254 012250 000000 MMABTO: HALT ;MEMORY MANAGEMENT TRAP
2255 012252 000002 RTI ;CAUSED THE ABORT

```

```

;BACKGROUND TEST FOR INTERRUPTS
2256
2257
2258 012254 052737 002000 001122 XWAIT: BIS #BIT10,FLAG2 ;WAITING IN BACKGROUND TEST
2259 012252 012737 070000 012370 MOV #70000,NPRCNT ;SETUP TIMEOUT COUNTER
2260 012270 012701 012373 MOV #NPR1+1,R1 ;SETUP WAIT LOOP
2261 012274 112711 000200 MOVB #200,(R1)
2262 012300
2263 012300 105421 2$: NEGB (R1)+
2264 012302 105441 NEGB -(R1)
2265 012304 105421 NEGB (R1)+
2266 012306 105441 NEGB -(R1)
2267 012310 105421 NEGB (R1)+
2268 012312 105441 NEGB -(R1)
2269 012314 105421 NEGB (R1)+
2270 012316 105441 NEGB -(R1)
2271 012320 105421 NEGB (R1)+
2272 012322 105441 NEGB -(R1)
2273 012324 105421 NEGB (R1)+
2274 012326 105441 NEGB -(R1)
2275 012330 105421 NEGB (R1)+
2276 012332 105441 NEGB -(R1)
2277 012334 105421 NEGB (R1)+
2278 012336 105441 NEGB -(R1)
2279 012340 102401 BVS 1$
2280 012342 000000 HALT ;ARITHMETIC OPERATION FAILED RUN DIAG
2281 012344 005337 012370 1$: DEC NPRCNT
2282 012350 001353 BNE 2$
2283 012352 104054 HLT !DA!WC!DS ;TIMED OUT NO INTERRUPT
2284 012354 000137 001234 JMP @#BEGIN
2285 012360 042737 002000 001122 NPRRET: BIC #BIT10,FLAG2 ;CLEAR BKGROUND FLG
2286 012366 000207 RTS PC
2287 012370 000000 NPRCNT: 0
2288 012372 000000 NPR1: 0
2289 ;CLEAR ERROR TABLE
2290
2291 012374 012704 000020 ERRCL: MOV #20,R4 ;CLEAR
2292 012400 012703 017014 MOV #ERTAB,R3 ;ERROR
2293 012404 005023 1$: CLR (R3)+ ;TABLE
2294 012406 005304 DEC R4 ;DONE YET?
2295 012410 001375 BNE 1$ ;NO
2296 012412 005037 001004 CLR PCNT ;CLEAR
2297 012416 005037 001006 CLR PCNT+2 ;PASS COUNT
2298 012422 005037 001124 CLR DROP ;CLEAR ALL DROPPED DRIVES
2299 012426 000205 RTS R5 ;RETURN

```

```

2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317 012430 012706 000500
2318 012434 104402 000510
2319 012440 104420
2320 012442 012637 001160
2321 012446 004737 006722
2322 012452 004737 006466
2323 012456 104426
2324 012460 004737 013234
2325 012464 005037 001134
2326 012470 012737 012712 000024
2327 012476 012737 000340 000026
2328 012504 012737 000161 001172
2329 012512 104416
2330 012514 004737 011712
2331 012520 032737 001000 001126
2332 012526 001406
2333 012530 104006
2334 012532 012777 177777 166310
2335 012540 005077 166302
2336 012544 004737 006776
2337 012550 000755
2338 012552 000744

```

```

;RH11 POWER FAIL TEST #1
;THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 244. THE PROGRAM
;WRITE THE COMPLETE DISK WITH A 125252 PATTERN. THE PROGRAM WILL THEN
;TELL OPERATOR TO POWER DOWN. UNTIL THE POWER FAIL, THE PROGRAM WILL
;CONTINUE WRITING THE SAME PATTERN ON THE DISK.
;WHEN POWER FAIL OCCURS THE TRANSFER IS ABORTED
;AND THE PROGRAM HALTS. THE OPERATOR SHOULD
;NOW TURN POWER BACK ON. THE PROGRAM RESTARTS AND CHECKS FOR WRITE ERRORS.
;ONLY ONE ERROR IS ACCEPTABLE. THAT ERROR MAY BE AN OPI (BIT13 RSER)OR A DCK
;(BIT 15 RSER). IF THESE ARE THE ONLY ERRORS THAT OCCUR, THE DRIVE IS OK.
;IF NO ERRORS OCCUR, THE PROGRAM WILL TYPE OUT "OK".
;THE PROGRAM WILL THEN TELL YOU WHEN TO POWER DOWN AGAIN

;***ONLY ONE ERROR IS CONSIDERED ACCEPTABLE***
;NOTE: ALL DRIVES ON THE SYSTEM SHOULD BE POWERED OFF EXCEPT
;THE DRIVE UNDER TEST. *****

```

```

PFT1:  MOV      #500,SP      ;SET UP STACK
        TYPE    ,LOADSW
        RDOCT
        MOV     (SP)+,UNNUM
2$:     JSR     PC,VECTRR    ;SETUP INT VECTOR
        JSR     PC,FNDTYP   ;TST FOR RS03 OR 04
PFWATT: CLRDV              ;CLEAR ALL REG
        JSR     PC,POWFAL   ;WRITE 125252 ON DISK
PFWAT:  CLR     DMA
        MOV     #DOWN,24    ;SET UP POWER FAIL VEC.
        MOV     #340,26
MYBYWR: MOV     #161,CMD    ;WRITE WITH I/E
        DKCMD              ;DO IT
        JSR     PC,WATT     ;WAIT FOR INTERRUPT
3$:     BIT     #BIT9,FLAG  ;ANY ERRORS?
        BEQ    1$          ;NO
        HLT    !DA!DB
        MOV     #-1,RSAS   ;CLEAR ALL
        CLR    RRSER      ;ERRORS
1$:     JSR     PC,DISBUF   ;SET UP NEW DISK BUFFER
        BR     MYBYWR
        BR     PFWAT

```


E05

CZRSCF RH11-RS03-RS03/LA-RS04
CZRSCF.P11 16-DEC-77 15:10

DATA AND RELIABILITY TEST
TST4 TEST FOR MULTI DISK MODE

MACY11 30(1046) 20-DEC-77 15:36 PAGE 57

SEQ 0056

```

2339 012554 012737 012562 001152 UPCHK: MOV #1$,HRDER ;RETURN HERE IF WRONG DRIVE INTERRUPTS
2340 012562 005037 001134 1$: CLR DMA
2341 012566 104426 CLRDV ;INIT DRIVE
2342 012570 013737 001066 177776 CHKDAT: MOV PRIORITY,PS
2343 012576 012737 000151 001172 MOV #151,CMD ;WRITECHECK WITH I/E
2344 012604 104416 DKCMD ;DO IT
2345 012606 013737 001066 177776 MOV PRICRITY,PS
2346 012614 004737 011712 JSR PC,WATT ;WAIT FOR INTERRUPT
2347 012620 032737 001000 001126 3$: BIT #BIT9,FLAG ;ANY ERRORS?
2348 012626 001411 BEQ 1$ ;NO
2349 012630 104006 HLT !DB!DA
2350 012632 052737 100000 001122 BIS #BIT15,FLAG2 ;SET ERROR FLAG
2351 012640 005077 166202 CLR JRSER ;CLEAR ALL
2352 012644 012777 177777 166176 MOV #-1,ARSAS ;ERRORS
2353 012652 004737 006776 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
2354 012656 000744 BR CHKDAT
2355 012660 005737 001122 TST FLAG2 ;ANY ERRORS?
2356 012664 100405 BMI 2$ ;YES
2357 012666 104402 012672 TYPE .+2 ;.ASCIZ <15><12>"OK"
2358 012700 042737 100000 001122 2$: BIC #BIT15,FLAG2 ;CLEAR ERROR FLAG
2359 012706 000137 012456 JMP PFWATT ;GO WAIT FOR ANOTHER
2360 ;POWER FAIL
2361 ;POWER DOWN ROUTINE - ABORT DISK AND HALT
2362
2363
2364 012712 012737 012722 000024 DOWN: MOV #UPP,24 ;SET POWER FAIL VECTOR
2365 012720 000000 HALT
2366
2367 012722 012737 012712 000024 UPP: MOV #DOWN,24
2368 012730 012706 000500 MOV #500,SP
2369 012734 013777 001160 166072 MOV UNNUM,ARSAS2 ;GET UNIT #
2370 012742 032777 000200 166074 1$: BIT #BIT7,ARSDS ;WAIT FOR DRIVE READY
2371 012750 001774 BEQ 1$
2372 012752 000137 012554 JMP UPCHK ;GO CHECK DISK

```

F05

```

2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389 012756 012706 000500
2390 012762 042737 001000 001122
2391 012770 012737 013012 001152
2392 012776 104426
2393 013000 004737 006722
2394 013004 004737 013234
2395 013010 000401
2396 013012 104426
2397 013014 005037 001134
2398 013020 012737 013164 000024
2399 013026 012737 000340 000026
2400 013034 013737 001066 177776
2401 013042 012737 000151 001172
2402 013050 104416
2403 013052 004737 011712
2404 013056 032737 001000 001126
2405 013064 001411
2406 013066 104002
2407 013070 052737 100000 001122
2408 013076 005077 165744
2409 013102 012777 177777 165740
2410 013110 004737 006776
2411 013114 000747
2412 013116 032737 001000 001122
2413 013124 001733
2414 013126 005737 001122
2415 013132 100405
2416 013134 104402 013140
2417 013146 042737 100000 001122
2418 013154 042737 001000 001122
2419 013162 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      #PWRFL1,HRDR ;RETURN HERE IF WRONG DRIVE INT.
      CLRDV   ;INIT DRIVE
      JSR     PC,VECTRR    ;SETUP INT VECTOR
PWRFL2: JSR     PC,POWFAL  ;WRITE 125252 ON DISK
      BR      PWRFL       ;WRITE CHECK
      CLRDV   ;INIT DRIVE
PWRFL1: CLRDV
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     @RSER       ;CLEAR ALL
      MOV     #-1,@RSAS   ;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

```

;ROUTINE TO ABORT DISK DURING POWER FAIL

```

2420
2421
2422 013164 012737 013174 000024 PWRDN: MOV #PWRUP,24 ;SET UP RESTART
2423 013172 000000 HALT
2424
2425 013174 012737 013164 000024 PWRUP: MOV #PWRDN,24 ;RESET POWER FAIL VECTOR
2426 013202 012706 000500 MOV #S0C,SP
2427 013206 013777 001160 165620 MOV UNNUM,DRSCS2 ;GET UNIT #
2428 013214 052737 001000 001122 BIS #BIT9,FLAG2 ;SET POWER FAIL BIT
2429 013222 032777 000200 165614 1$: BIT #BIT7,DRSDS ;WAITING FOR
2430 013230 001774 BEQ 1$ ;DRIVE READY
2431 013232 000667 BR PWRF1 ;GO CHECK DISK
2432
2433
2434
2435

```

;ROUTINE TO WRITE THE COMPLETE DISK
;WITH 125252 PATTERN
;WRITE CHECK AND REPORT ERRORS IF THEY OCCUR
;REPORT "OK" AT COMPLETION

```

2436
2437
2438
2439 013234 012737 000020 001136 POWFAL: MOV #20,PATNU ;SET UP PATTERN
2440 013242 042737 000004 001126 BIC #BIT2,FLAG ;CLEAR XFER MODE FLAG
2441 013250 052737 010000 001122 BIS #BIT12,FLAG2
2442 013256 005037 001134 CLR DMA
2443 013262 012737 020000 017360 MOV #20000,OUTBUF ;GET STARTING ADDR FOR BUF
2444 013270 012737 020000 001116 MOV #20000,VADDR
2445 013276 012737 010000 001144 MOV #10000,SWRDCT ;SETUP WORD COUNT
2446 013304 013737 001144 001130 MOV SWRDCT,WRDCT
2447 013312 005037 001114 CLR AOB1 ;A PORT ONLY
2448 013316 013737 017360 001140 MOV OUTBUF,BUF ;SET UP CURRENT ADDRESS
2449 013324 004537 007644 JSR RS,PASEL ;GENERATE DATA BUFFER
2450 013330 012737 000161 001172 WRDNW: MOV #161,CMD ;WRITE WITH I/E
2451 013336 104416 DKCMD ;DO IT
2452 013340 004737 011712 JSR PC,WATT ;WAIT FOR INTERRUPT
2453 013344 012737 000151 001172 2$: MOV #151,CMD ;WRITECHECK I/E
2454 013352 104416 DKCMD ;DO IT
2455 013354 004737 011712 JSR PC,WATT ;WAIT FOR INTERRUPT
2456 013360 032737 001000 001126 4$: BIT #BIT9,FLAG ;ANY ERRORS?
2457 013366 001402 BEQ 1$ ;NO
2458 013370 104006 HLT !DB!DA ;YES
2459 013372 000000 HALT ;CAN NOT WRITE WITHOUT ERROR
2460 013374 004737 006776 1$: JSR PC,DISBUF ;SET UP NEW DISK BUFFER
2461 013400 000753 BR WRDNW ;WRITE NEW BUFFER
2462 013402 104402 000721 TYPE PDOWN
2463 013406 000207 RTS PC

```

H05

CZRSCF RH11-R503-R503/LA-R504 DATA AND RELIABILITY TEST
 CZRSCF.P11 16-DEC-77 15:10

TST4 TEST FOR MULTI DISK MODE

MACY11 30(1046) 20-DEC-77 15:36 PAGE 60

SEQ 0059

2464	013410	032777	000010	165410	OUT:	BIT	#BIT3,2SWR	:TYPEOUT ERROR COUNT?
2465	013416	001526				BEQ	1\$:NO
2466	013420	005004				CLR	R4	:CLEAR UNIT #
2467	013422	005003				CLR	R3	
2468	013424	053737	001124	001162		BIS	DROP,UNITSV	:RESTORE ALL DRIVES
2469	013432	013737	001162	001222		MOV	UNITSV,WORK	:GET UNITS ON SYSTEM
2470	013440	012705	000401			MOV	#40,R5	:SETUP TEST FOR UNITS
2471	013444	030537	001222		4\$:	BIT	R5,WORK	:IS THIS UNIT ON SYS
2472	013450	001006				BNE	2\$:YES
2473	013452	005204			5\$:	INC	R4	:INC UNIT #
2474	013454	010403				MOV	R4,R3	:SAVE UNIT #
2475	013456	000241				CLC		
2476	013460	006105				ROL	R5	:GET NEXT DRIVE
2477	013462	103501				BCS	3\$:DONE
2478	013464	000767				BR	4\$:FIND NEXT DRIVE
2479	013466	104402	000510		2\$:	TYPE	LOADSW	
2480	013472	010446				MOV	R4,-(6)	:PUT R4 ON STACK
2481	013474	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2482	013476	004737	013754			JSR	PC,GETERR	:GET ERROR COUNT
2483	013502	010304				MOV	R3,R4	:RESTORE UNIT #
2484	013504	104402	013510			TYPE	,+2	:.ASCIZ <15><12>
2485	013514	104402	000564			TYPE	,WRERR	
2486	013520	104402	013524			TYPE	,+2	:.ASCIZ "S "
2487	013530	013746	001202			MOV	WRITER,-(6)	:PUT WRITER ON STACK
2488	013534	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2489	013536	104402	013542			TYPE	,+2	:.ASCIZ <15><12>
2490	013546	104402	000607			TYPE	,RDERR	
2491	013552	104402	013556			TYPE	,+2	:.ASCIZ "S "
2492	013562	013746	001206			MOV	READER,-(6)	:PUT READER ON STACK
2493	013566	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2494	013570	104402	013574			TYPE	,+2	:.ASCIZ <15><12>
2495	013600	104402	000574			TYPE	,WCERR	
2496	013604	104402	013610			TYPE	,+2	:.ASCIZ "S "
2497	013614	013746	001204			MOV	WCERR,-(6)	:PUT WCERR ON STACK
2498	013620	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2499	013622	104402	013626			TYPE	,+2	:.ASCIZ <15><12>"COMPARE ERRS "
2500	013646	013746	001210			MOV	COMERR,-(6)	:PUT COMERR ON STACK
2501	013652	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2502	013654	104402	013660			TYPE	,+2	:.ASCIZ <15><12>
2503	013664	000672				BR	5\$:GET NEXT DRIVE
2504	013666	043737	001124	001162	3\$:	BIC	DROP,UNITSV	:REDROP DRIVES
2505	013674	062706	000002		1\$:	ADD	#2,SP	:RESTORE SP DUE TO JMP EXIT FROM JSR ROUTINE
2506	013700	005137	001114			COM	A0B1	:SET A OR B PORT FLAG
2507	013704	032777	000040	165114		BIT	#BITS,2SWR	:TYPEOUT PASS COUNT?
2508	013712	001035				BNE	DONE	:NO
2509	013714	104402	013720			TYPE	,+2	:.ASCIZ <15><12>"END PASS "
2510	013734	013746	001006			MOV	PCNT+2,-(6)	:PUT PCNT+2 ON STACK
2511	013740	104406				TYPES		:TYPE STACK IN OCTAL - SUPRESS
2512	013742	104402	013746			TYPE	,+2	:.ASCIZ <15><12>
2513	013752	000415				BR	DONE	

```

2514 013754 006304          GETERR: ASL      R4          ;GET LOC IN
2515 013756 006304          ASL      R4          ;ERR TABLE
2516 013760 062704 017014  ADD      #ERTAB,R4
2517 013764 112437 001202  MOVB    (R4)+,WRITER ;GET WRITE ERRS
2518 013770 112437 001206  MOVB    (R4)+,READER ;GET READ ERRS
2519 013774 112437 001204  MOVB    (R4)+,WCERR  ;GET WRITE CK ERRS
2520 014000 112437 001210  MOVB    (R4)+,COMERR ;GET COMPARE ERRS
2521 014004 000207  RTS     PC
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
014006 104400          .SBTTL      $DONE - BELL AND SCOPE ROUTINE
014010 062737 000001 001006  DONE:     SCOPE      ;TERMINATIONG SCOPE FOR LOOPING
014016 005537 001004          ADD      #1,PCNT+2  ;ADD 1 TO THE PASS COUNT
014022 013700 000042  4$:      ADC      PCNT   ;MAKE IT DOUBLE PREC.
014026 001405          MOV      #42,R0    ;GET MONITOR ADDRESS
014030 000005          BEQ     $END1     ;IF NONE
014032 004710          RESET
014034 000240 000240 000240  $ENDAD:  JSR      7,(0) ;GO TO MONITOR
014042 000137 003154  $END1:   JMP      240,240,240 ;SAVE ROOM FOR ACT11
014046 000000          .TBIT:    0        ;T BIT FLAG
014050 012702 000001  .LOGW:   MOV      #1,R2 ;LOG WRITE ERR
014054 005003  CLIND:   CLR      R3    ;CLEAR INDEX FOR TABLE
014056 000413          BR      ADDR
014060 012702 000400  .LOGR:   MOV      #400,R2 ;LOG WRITE ERR
014064 000773          BR      CLIND
014066 012702 000001  .LOGWC:  MOV      #1,R2 ;LOG WRITC CK ERR
014072 012703 000002  SETIND:  MOV      #2,R3 ;SET INDEX FOR NEXT WD
014076 000403          BR      ADDR
014100 012702 000400  .LOGC:   MOV      #400,R2 ;LOG COMPARE ERR
014104 000772          BR      SETIND
014106 005737 001200  ADDR:    TST      LOPCNT ;1ST TIME ERROR?
014112 001014          BNE     1$        ;NO DO NOT COUNT IT
014114 013704 001160  MOV      UNNUM,R4  ;GET UNIT #
014120 006304          ASL     R4        ;GET
014122 006304          ASL     R4        ;POSITION IN
014124 060304          ADD     R3,R4    ;ERR TABLE
014126 060264 017014  ADD     R2,ERTAB(R4) ;TO ADD ERROR
014132 004737 014146  JSR     PC,PRNT   ;TYPEOUT?
014136 001402          BEQ     1$        ;YES
014140 004737 014734  JSR     PC,DRP    ;SHOULD I DROP DRIVE?
014144 000002  1$:      RTI
014146 032777 020000 164652  PRNT:   BIT      #BIT13,$SWR ;INHIBIT TYPEOUT?
014154 000207  RTS     PC

```

2565	014156	052737	000004	001122	RSREG:	BIS	#BIT2,FLAG2	;SET ERROR FLAG
2566	014164	005737	016002			TST	.HLTCT	;SHOULD WE TYPE GOOD AND BAD
2567	014170	001017				BNE	8\$;NO
2568	014172	104402	014176			TYPE	+2	;ASCIZ "BAD="
2569	014204	010046				MOV	BAD,-(6)	;PUT BAD ON STACK
2570	014206	104404				TYPE0		;TYPE STACK IN OCTAL
2571	014210	104402	014214			TYPE	+2	;ASCIZ "GOOD="
2572	014224	010146				MOV	GOOD,-(6)	;PUT GOOD ON STACK
2573	014226	104404				TYPE0		;TYPE STACK IN OCTAL
2574	014230				8\$:			
2575	014230	104402	014234			TYPE	+2	;ASCIZ "CS1="
2576	014242	017746	164564			MOV	ARSCS1,-(6)	;PUT ARSCS1 ON STACK
2577	014246	104404				TYPE0		;TYPE STACK IN OCTAL
2578	014250				1\$:			
2579	014250	104402	014254			TYPE	+2	;ASCIZ "ER="
2580	014262	017746	164560			MOV	ARSER,-(6)	;PUT ARSER ON STACK
2581	014266	104404				TYPE0		;TYPE STACK IN OCTAL
2582	014270				2\$:			
2583	014270	104402	014274			TYPE	+2	;ASCIZ "CS2="
2584	014302	017746	164526			MOV	ARSCS2,-(6)	;PUT ARSCS2 ON STACK
2585	014306	104404				TYPE0		;TYPE STACK IN OCTAL
2586	014310	032737	000200	016002		BIT	#200,.HLTCT	;PRINT SECOND SET ?
2587	014316	001112				BNE	SEEC	;YES
2588	014320	032737	000100	016002		BIT	#AS,.HLTCT	;PRINT ER ?
2589	014326	001410				BEQ	3\$;NO
2590	014330	104402	014334			TYPE	+2	;ASCIZ "AS="
2591	014342	017746	164502			MOV	ARSAS,-(6)	;PUT ARSAS ON STACK
2592	014346	104404				TYPE0		;TYPE STACK IN OCTAL
2593	014350	032737	000020	016002	3\$:	BIT	#BA,.HLTCT	;PRINT BUS ASSRESS
2594	014356	001410				BEQ	4\$;NO
2595	014360	104402	014364			TYPE	+2	;ASCIZ "BA="
2596	014372	017746	164442			MOV	ARSBA,-(6)	;PUT ARSBA ON STACK
2597	014376	104404				TYPE0		;TYPE STACK IN OCTAL
2598	014400	032737	000004	016002	4\$:	BIT	#DA,.HLTCT	;PRINT DA ?
2599	014406	001410				BEQ	5\$;NO
2600	014410	104402	014414			TYPE	+2	;ASCIZ "DA="
2601	014422	017746	164414			MOV	ARSDA,-(6)	;PUT ARSDA ON STACK
2602	014426	104404				TYPE0		;TYPE STACK IN OCTAL
2603	014430	032737	000010	016002	5\$:	BIT	#WC,.HLTCT	;PRINT WC?
2604	014436	001410				BEQ	6\$;NO
2605	014440	104402	014444			TYPE	+2	;ASCIZ "WC="
2606	014452	017746	164360			MOV	ARSWC,-(6)	;PUT ARSWC ON STACK
2607	014456	104404				TYPE0		;TYPE STACK IN OCTAL
2608	014460	032737	000040	016002	6\$:	BIT	#DS,.HLTCT	;DRIVE STATUS
2609	014466	001410				BEQ	9\$;NO
2610	014470	104402	014474			TYPE	+2	;ASCIZ "DS="
2611	014502	017746	164336			MOV	ARSDS,-(6)	;PUT ARSDS ON STACK
2612	014506	104404				TYPE0		;TYPE STACK IN OCTAL
2613	014510	032737	000002	016002	9\$:	BIT	#DB,.HLTCT	;PRINT DATA BUFFER

2614	014516	001461				BEG	PTDONE	:	NO
2615	014520	104402	014524			TYPE	.+2	:	.ASCIZ " DB="
2616	014532	017746	164316			MOV	ARSDB,-(6)	:	PUT ARSDB ON STACK
2617	014536	104404				TYPEO		:	TYPE STACK IN OCTAL
2618	014540	000137	014662			JMP	PTDONE	:	GET OUT
2619	014544	042737	000200	016002	SEEC:	BIC	#200,.HLTCT	:	CLEAR COMMON BIT
2620	014552	032737	000240	016002		BIT	#DT,.HLTCT	:	PRINT DRIVE TYPE?
2621	014560	001410				BEG	10\$:	NO
2622	014562	104402	014566			TYPE	.+2	:	.ASCIZ " DT="
2623	014574	017746	164260			MOV	ARSDT,-(6)	:	PUT ARSDT ON STACK
2624	014600	104404				TYPEO		:	TYPE STACK IN OCTAL
2625	014602	032737	000220	016002	10\$:	BIT	#MR,.HLTCT	:	PRINT MN?
2626	014610	001410				BEG	11\$:	NO
2627	014612	104402	014616			TYPE	.+2	:	.ASCIZ " MR="
2628	014624	017746	164226			MOV	ARSMR,-(6)	:	PUT ARSMR ON STACK
2629	014630	104404				TYPEO		:	TYPE STACK IN OCTAL
2630	014632	032737	000204	016002	11\$:	BIT	#LA,.HLTCT	:	PRINT LA?
2631	014640	001410				BEG	PTDONE	:	NO
2632	014642	104402	014646			TYPE	.+2	:	.ASCIZ " LA="
2633	014654	017746	164172			MOV	ARSLA,-(6)	:	PUT ARSLA ON STACK
2634	014660	104404				TYPEO		:	TYPE STACK IN OCTAL
2635	014662	032737	010000	001122	PTDONE:	BIT	#BIT12,FLAG2	:	POWER FAIL TEST?
2636	014670	001111				BNE	RETT	:	YES
2637	014672	104402	014676			TYPE	.+2	:	.ASCIZ <15><12>"PASS "
2638	014706	013746	001006			MOV	PCNT+2,-(6)	:	PUT PCNT+2 ON STACK
2639	014712	104406				TYPES		:	TYPE STACK IN OCTAL - SUPRESS
2640	014714	032777	001000	164104		BIT	#BIT9,ASWR	:	LOOPING ON ERROR?
2641	014722	001404				BEG	DRP	:	NO
2642	014724	104402	014730			TYPE	.+2	:	.ASCIZ <15><12>
2643	014734	032777	000001	164064	DRP:	BIT	#BIT0,ASWR	:	DROP DRIVE?
2644	014742	001464				BEG	RETT	:	NO
2645	014744	013704	001160			MOV	UNNUM,R4	:	GET UNIT #
2646	014750	004737	013754			JSR	PC,GETERR	:	GET ERRORS
2647	014754	063737	001202	001206		ADD	WRITER,READER	:	ADD THE ERRORS
2648	014762	063737	001206	001204		ADD	READER,WCERR	:	
2649	014770	063737	001204	001210		ADD	WCERR,COMERR	:	
2650	014776	022737	000023	001210		CMP	#23,COMERR	:	DROP DRIVE?
2651	015004	103043				BHIS	RETT	:	NO
2652	015006	053737	001164	001124		BIS	UNCMP,DROP	:	DROP DRIVE
2653	015014	104402	015020			TYPE	.+2	:	.ASCIZ <15><12>"DROPPED UNIT "
2654	015040	013746	001160			MOV	UNNUM,-(6)	:	PUT UNNUM ON STACK
2655	015044	104406				TYPES		:	TYPE STACK IN OCTAL - SUPRESS
2656	015046	104402	000636			TYPE	CRLF	:	
2657	015052	113703	001124			MOVB	DROP,R3	:	GET DROPPED UNITS
2658	015056	113704	001162			MOVB	UNITSV,R4	:	GET ALL DRIVES
2659	015062	020304				CMP	R3,R4	:	ALL DRIVES DROPPED?
2660	015064	001003				BNE	2\$:	NO
2661	015066	000000				HALT		:	NO MORE DRIVES
2662	015070	000137	001234			JMP	ASBEGIN	:	RESTART TEST
2663	015074	032737	100000	001126	2\$:	BIT	#BIT15,FLAG	:	DID OPERATOR SELECT PATTERN
2664	015102	001002				BNE	3\$:	YES
2665	015104	005037	001136			CLR	PATNU	:	NO CLEAR IT
2666	015110	000137	006014		3\$:	JMP	ASEXTPPR	:	GET NEXT DRIVE
2667	015114	000207			RETT:	RTS	PC	:	

```

2668      :ROUTINE TO RESTORE LOADER
2669 015116 013705 015142  RLDR:  MOV      LDR1,R5      ;GET FIRST ADDRESS OF WHERE LOADER IS
2670      :TO BE RESTORED
2671 015122 012704 017446      MOV      #17446,R4      ;ADDRESS WHERE LOADER IS STORED
2672 015126 012702 000155      MOV      #155,R2       ;WORD COUNT
2673 015132 012425      1$:  MOV      (R4)+,(R5)+  ;RESTORE
2674 015134 005302      DEC      R2
2675 015136 001375      BNE     1$
2676 015140 000000      HALT
2677 015142 017446  LDR1:  .WORD   17446      ;DONE
2678      :FIRST ADDRESS WHERE LOADERS ARE SAVED
2679      172100
2680      000114  PARCSR=172100
2681 015144 012737 015236 000114  .MAMK:  MOV      #,PARSRV,#PARVEC
2682 015152 012737 000340 000116  MOV      #340,#PARVEC+2 ;SET PRI LEVEL TO 7
2683 015160 013746 000004      MOV      @#4,-(SP)      ;SAVE CURRENT ERROR VECTOR
2684 015164 013746 000006      MOV      @#6,-(SP)      ;SAVE PRIORITY LEVEL
2685 015170 012737 000006 000004      MOV      #6,@#4
2686 015176 012737 000002 000006      MOV      #RTI,@#6
2687 015204 012700 172100      MOV      #PARCSR,R0      ;GET FIRST CSR ADDR
2688 015210 012702 000001      MOV      #1,R2
2689 015214 012720 000001  1$:  MOV      #1,(R0)+      ;SET ACTION ENABLE IF AVAILABLE
2690 015220 006302      ASL     R2              ;SHIFT AVAILABILITY INDICATOR
2691 015222 103374      BCC     1$
2692 015224 012637 000006      MOV      (SP)+,@#6      ;RESTORE ERROR VECTOR PRIORITY
2693 015230 012637 000004      MOV      (SP)+,@#4      ;AND INTERRUPT VECTOR
2694 015234 000207      RTS     PC
2695      ;PARITY MEMORY TRAP
2696
2697 015236 104402 000736      .PARSRV:  TYPE     PAREER
2698 015242 052737 004000 001122  BIS     #BIT11,FLAG2    ;SET ERROR FLAG
2699 015250 032737 000010 001122  BIT     #BIT3,FLAG2    ;WERE WE COMPARING DURING ERROR?
2700 015256 001422      BEQ     13$            ;NO
2701 015260 104402 015264      TYPE     .+2          ;ASCIZ " ADDR="
2702 015274 005737 001212      TST     MMAVA         ;IS MEM MGMT ON?
2703 015300 001406      BEQ     12$            ;NO
2704 015302 013746 177776      MOV     PS,-(6)        ;GET PS
2705 015306 013746 001150      MOV     SAVE,-(6)      ;GET VIRTUAL ADDR
2706 015312 104412      TYPEA   13$          ;CONVERT TO PHY AND TYPE
2707 015314 000403      BR      13$          ;CONT
2708 015316 013746 001150  12$:  MOV     SAVE,-(6)      ;GET ADDR
2709 015322 104406      TYPES   13$          ;TYPE IT
2710 015324 032777 100000 163474  13$:  BIT     #BIT15,@SWR    ;HALT ON ERROR?
2711 015332 001401      BEQ     1$            ;NO
2712 015334 000000      HALT    ;YES
2713 015336 012706 000500  1$:  MOV     #500,SP        ;RESET STACK
2714 015342 000137 003170      JMP     EXMFLG        ;RESTART TEST

```


M05

```

2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725 015346 010446
2726 015350 010546
2727 015352 017605 000004
2728 015356 032705 177400
2729 015362 001002
2730 015364 016605 000004
2731 015370 105715
2732 015372 001423
2733 015374 122715 000012
2734 015400 001012
2735 015402 113704 001015
2736 015406 113777 001014 163410
2737 015414 105777 163376
2738 015420 100375
2739 015422 005304
2740 015424 001370
2741 015426 112577 163372
2742 015432 105777 163360
2743 015436 100375
2744 015440 000753
2745 015442 017646 000004
2746 015446 062766 000002 000006
2747 015454 022666 000004
2748 015460 001006
2749 015462 062705 000002
2750 015466 042705 000001
2751 015472 010566 000004
2752 015476 012605
2753 015500 012604
2754 015502 000002
  
```

```

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

.TYPE: MOV R4,-(6) ;SAVE R4
MOV R5,-(6) ;SAVE R5
MOV @4(6),R5 ;GET ADDRESS TO BE TYPED
BIT #177400,R5 ;IS IT A TYPED?
BNE 1$ ;NO
MOV 4(6),R5 ;GET ADDRESS OF CHARACTER
1$: TSTB (R5) ;TERMINATOR?
BEQ 2$ ;GET OUT IF SO
CMPB #12,(R5) ;IS THE CHAR A LINE FEED
BNE 4$ ;NO - GET OUT
MOVB FILCHR+1,R4 ;GET THE FILL COUNT
5$: MOVB FILCHR,@TPB ;TYPE A FILLER
TSTB @TPB ;DONE YET?
BPL -4 ;NO - WAIT
DEC R4 ;DEC COUNT
5$ ;LOOP UNTIL 0
MOVB (R5)+,@TPB ;LOAD AND TYPE THE CHARACTER
4$: TSTB @TPB ;IS THE PRINTER READY
BPL -4 ;WAIT UNTIL IT IS
BR 1$ ;GET THE NEXT CHARACTER
2$: MOV @4(6),-(6) ;GET ADDRESS TO BE TYPED
ADD #2,6(6) ;ADD 2 TO THE ADDRESS
CMP (6)+,4(6) ;IS IT .+2?
BNE 3$ ;NO
ADD #2,R5 ;ADD 2 TO THE ADDRESS
BIC #1,R5 ;BACK UP TO AN EVEN BYTE
MOV R5,4(6) ;RESTORE ADDRESS
3$: MOV (6)+,R5 ;RESTORE R5
MOV (6)+,R4 ;RESTORE R4
RTI ;RETURN
  
```

```

2755 .SBTTL $SCOPE - SCOPE LOOP HANDLER
2756
2757 ; THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
2758 ; LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
2759 ; "SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
2760 ; RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"
2761
2762 015504 104442 .SCOPE: KBDIN ; GO CHECK FOR ↑G
2763 015506 032777 040000 163312 BIT #SW14, @SWR ; LOOP ON TEST?
2764 015514 001045 BNE .KIT ; LOOP ON TEST IS SET
2765 015516 000416 BR 3$ ; SKIP - NOP FOR XOR TESTER
2766 015520 013746 000004 MOV @#4, -(6) ; PUSH @#4 ON STACK
2767 015524 012737 015544 000004 MOV #4$,@#4 ; SET FOR TIMEOUT
2768 015532 005737 177060 TST @#177060 ; ERROR ON XOR?
2769 015536 012637 000004 MOV (6)+,@#4 ; POP STACK INTO @#4
2770 015542 000422 BR .SVLAD ; NO ERROR - GO TO NEXT TEST
2771 015544 022626 4$: CMP (6)+,(6)+ ; CLEAR STACK
2772 015546 012637 000004 MOV (6)+,@#4 ; POP STACK INTO @#4
2773 015552 000426 BR .KIT ; ERROR - LOOP ON TEST
2774 015554 032777 004000 163244 3$: BIT #SW11,@SWR ; KILL ITERATIONS
2775 015562 001012 BNE .SVLAD ; YES - KILL ITERATIONS
2776 015564 105737 001001 TSTB ICNT+1 ; FIRST ONE?
2777 015570 001404 BEQ 2$ ; BRANCH IF FIRST
2778 015572 123737 015656 001001 CMPB TIMES, ICNT+1 ; DONE?
2779 015600 003013 BGT .KIT ; BRANCH IF NOT
2780 015602 112737 000001 001001 2$: MOVB #1, ICNT+1 ; FIRST ITERATION
2781 015610 105237 001000 .SVLAD: INCB ICNT ; COUNT TEST NUMBERS
2782 015614 011637 001010 MOV (6), LAD ; SAVE LOOP ADDRESS
2783 015620 013777 001000 163202 MOV ICNT, @DISPLAY ; DISPLAY TEST NO. AND ITERATION COUNT
2784 015626 000002 RTI ; RETURN
2785
2786 015630 105237 001001 .KIT: INCB ICNT+1 ; INC THE ITERATION COUNT
2787 015634 013777 001000 163166 .OVER: MOV ICNT, @DISPLAY ; SET UP DISPLAY
2788 015642 005737 001010 TST LAD ; FIRST ONE?
2789 015646 001760 BEQ .SVLAD ; YES
2790 015650 013716 001010 MOV LAD, (6) ; FUDGE RETURN ADDRESS
2791 015654 000002 RTI ; FIXES PS
2792
2793 015656 000001 TIMES: 1 ; RUN 1 TIMES

```

\$HLT - HLT ROUTINE (ERROR TYPEOUT)

```

2794
2795
2796
2797
2798
2799
2800
2801
2802 015660 005237 001002
2803 015664 104442
2804 015666 032777 020000 163132
2805 015674 001025
2806 015676 104402 015702
2807 015706 011637 001012
2808 015712 162737 000002 001012
2809 015720 117737 163066 016002
2810 015726 013746 001012
2811 015732 104404
2812 015734 104402 015740
2813 015744 004737 014156
2814 015750 005777 163052
2815 015754 100001
2816 015756 000000
2817 015760 032777 001000 163040
2818 015766 001003
2819 015770 105037 001001
2820 015774 000002
2821 015776 000137 015630
2822
2823 016002 000000

```

```

.SBTTL          $HLT - HLT ROUTINE (ERROR TYPEOUT)
;THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
;ADDRESS OF THE "HLT". IT ALSO COUNTS THE NUMBER OF ERRORS
;AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
;"HALT" ON ERROR, AND INHIBIT TYPEOUTS. AN OPTIONAL ARGUMENT
; (HLT+3) WILL BE PLACED IN ".HLTCT:" FOR ADITIONAL TYPEOUTS.
.HLT:  INC      ERRORS          ;INC THE ERROR COUNT
      KBDIN          ;GO CHECK FOR IG
      BIT      #SW13,@SWR      ;SKIP TYPEOUT IF SET
      BNE      2$          ;SKIP TYPEOUTS
      TYPE     .+2          ;.ASCIZ <15><12>
      MOV      (6),HLTADR      ;PUT ADDRESS OF INSTRUCTION ON STACK
      SUB      #2,HLTADR      ;FUDGE ADDRESS
      MOVB     @HLTADR,.HLTCT  ;GET HLT ARGUMENT
      MOV      HLTADR,-(6)    ;PUT HLTADR ON STACK
      TYPE0    TYPE          ;TYPE STACK IN OCTAL
      JSR      PC,RSREG      ;.ASCIZ ""
      TST      @SWR          ;GO TO USER ERROR ROUTINE
      BPL      .+4          ;HALT ON ERROR
      HALT          ;SKIP IF CONTINUE
      BIT      #SW9,@SWR      ;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
      .KIT          ;LOOP ON TEST UNTIL NO ERRORS
.HLTCT: 0          ;HLT ARGUMENT

```

```

2824 .SBTTL SOCTAL - OCTAL TYPEOUT ROUTINE
2825
2826 ; THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
2827 ; ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, OR TYPE THE
2828 ; 16 BITS. IT IS CALLED VIA THE TYOCT, TYPBIT, OR TYPOCS MACRO'S.
2829
2830 016004 012737 170101 016172 .TYPEB: MOV #170101,.PR ;SET BIT FLAG AND 16 CHARACTER COUNT
2831 016012 000411 .PTIT BR .PTIT ;NOW TYPE IT IN BIT FORM
2832 016014 112737 000001 016172 .TYPE0: MOVB #1,.PR ;SET ZERO FILL SWITCH
2833 016022 000402 BR .+6 ;SKIP
2834 016024 005037 016172 .TYPES: CLR .PR ;SUPPRESS LEADING ZERO'S
2835 016030 112737 177772 016173 MOVB #-6,.PR+1 ;SET COUNT
2836 016036
2837 016036 010446 .PTIT: MOV R4,-(6) ;PUSH R4 ON STACK
2838 016040 010546 MOV R5,-(6) ;PUSH R5 ON STACK
2839 016042 016605 000010 MOV 10(6),R5 ;GET THE DATA
2840 016046 012704 016174 MOV #.PR+2,R4 ;SET POINTER TO FIRST ASCII CHAR.
2841 016052 105014 CLRB (4) ;CLEAR FIRST BYTE
2842 016054 000411 BR .PRF ;ROTATE FIRST BIT
2843 016056 105014 .PRL: CLRB (4) ;CLEAR BYTE OF CHARACTER
2844 016060 032737 000100 016172 BIT #100,.PR ;BIT TYPING MODE?
2845 016066 001004 BNE .PRF ;YES - SKIP 2 ROTATES
2846 016070 006105 ROL R5 ;ROTATE BIT INTO C
2847 016072 106114 ROLB (4) ;PACK IT
2848 016074 006105 ROL R5 ;ROTATE BIT INTO C
2849 016076 106114 ROLB (4) ;PACK IT
2850 016100 006105 .PRF: ROL R5 ;ROTATE BIT INTO C
2851 016102 106114 ROLB (4) ;PACK IT
2852 016104 105714 TSTB (4) ;IS IT ZERO?
2853 016106 001402 BEQ .+6 ;SKIP INC
2854 016110 105237 016172 INCB .PR ;SET FILL SWITCH
2855 016114 105737 016172 TSTB .PR ;CHECK FILL SWITCH
2856 016120 001402 BEQ .+6 ;SKIP BITSET
2857 016122 152724 000060 BISB #'0,(4)+ ;MAKE INTO ASCII CHAR
2858 016126 105237 016173 INCB .PR+1 ;INC COUNT
2859 016132 001351 BNE .PRL ;REPEAT
2860 016134 022704 016174 CMP #.PR+2,R4 ;EMPTY BUFFER?
2861 016140 001002 BNE .+6 ;SKIP IF NOT
2862 016142 112724 000060 MOVB #'0,(4)+ ;LOAD 1 ZERO
2863 016146 105014 CLRB (4) ;NULL TERMINATOR
2864 016150 104402 016174 TYPE .PR+2 ;TYPE IT
2865 016154 012605 MOV (6)+,R5 ;POP STACK INTO R5
2866 016156 012604 MOV (6)+,R4 ;POP STACK INTO R4
2867 016160 016666 000002 000004 MOV 2(6),4(6) ;GET RID OF
2868 016166 012616 MOV (6)+,(6) ;DATA WORD
2869 016170 000002 RTI ;RETURN
2870
2871 016172 000012 .PR: .BLKW 12 ;COUNT, SWITCH, AND OUTPUT BUFFER

```

```

2872          .SBTTL          SPOWER - POWER DOWN AND UP ROUTINES
2873
2874          ; THIS IS THE POWER FAIL ROUTINE WHICH WILL SAVE ALL
2875          ; THE GENERAL REGISTERS AND USER DEFINED REGISTERS THEN
2876          ; WAIT FOR POWER TO GO DOWN AND BE RESTORED.
2877          ; IF THERE ISN'T ENOUGH TIME FOR SAVING ALL THE REGISTERS,
2878          ; THE PROGRAM WILL HALT AT '.ILLUP'.
2879
2880 016216 012777 016344 000126 .POWER: MOV      #.ILLUP, @.PUVEC ; SET FOR FAST UP
2881 016224 012777 000340 000122      MOV      #340, @.PUVECS+2 ; PRIO:7
2882 016232 010046          MOV      R0, -(6) ; PUSH R0 ON STACK
2883 016234 010146          MOV      R1, -(6) ; PUSH R1 ON STACK
2884 016236 010246          MOV      R2, -(6) ; PUSH R2 ON STACK
2885 016240 010346          MOV      R3, -(6) ; PUSH R3 ON STACK
2886 016242 010446          MOV      R4, -(6) ; PUSH R4 ON STACK
2887 016244 010546          MOV      R5, -(6) ; PUSH R5 ON STACK
2888 016246 010637 016350      MOV      SP, SAVR6 ; SAVE SP
2889 016252 012777 016262 000072      MOV      #.POWUP, @.PUVEC ; SET UP VECTOR
2890 016260 000000          HALT ; WAIT FOR PF
2891
2892 016262 013706 016350      .POWUP: MOV      SAVR6, SP ; GET SP
2893 016266 005001          CLR      R1 ; WAIT LOOP FOR THE TTY
2894 016270 005201          IS: INC      R1 ; WAIT FOR THE INC
2895 016272 001376          BNE     1$ ; OF WORD
2896 016274 012605          MOV      (6)+, R5 ; POP STACK INTO R5
2897 016276 012604          MOV      (6)+, R4 ; POP STACK INTO R4
2898 016300 012603          MOV      (6)+, R3 ; POP STACK INTO R3
2899 016302 012602          MOV      (6)+, R2 ; POP STACK INTO R2
2900 016304 012601          MOV      (6)+, R1 ; POP STACK INTO R1
2901 016306 012600          MOV      (6)+, R0 ; POP STACK INTO R0
2902 016310 012737 016216 000024      MOV      #.POWER, @#24 ; SET UP THE POWER DOWN VECTOR
2903 016316 012737 000340 000026      MOV      #340, @#26 ; PRIO:7
2904 016324 104402 016330          TYPE   +2 ; .ASCIZ <15><12>"POWER"
2905 016340 000137 006740          JMP     TIMUP ; JMP TO USER ADDRESS
2906
2907 016344 000000          .ILLUP: HALT ; THE POWER UP SEQUENCE WAS STARTED
2908 016346 000776          BR      .-2 ; BEFORE THE POWER DOWN WAS COMPLETE
2909
2910 016350 000000          .SAVR6: 0 ; PUT THE SP HERE
2911 016352 000024 000026      .PUVEC: 24, 26 ; POWER UP VECTOR

```

```

2912
2913
2914
2915
2916
2917
2918
2919 016356
2920 016356 010446
2921 016360 010546
2922 016362 016605 000012
2923 016366 016604 000010
2924 016372 016666 000006 000010
2925 016400 016666 000004 000006
2926 016406 016666 000002 000004
2927 016414 012616
2928 016416 010346
2929 016420 000305
2930 016422 006005
2931 016424 006005
2932 016426 006005
2933 016430 042705 177771
2934 016434 016505 016554
2935 016440 010403
2936 016442 042704 160000
2937 016446 000303
2938 016450 006003
2939 016452 006003
2940 016454 006003
2941 016456 006003
2942 016460 042703 177761
2943 016464 060305
2944 016466 011505
2945 016470 006305
2946 016472 006305
2947 016474 006305
2948 016476 006305
2949 016500 005003
2950 016502 006305
2951 016504 006103
2952 016506 006305
2953 016510 006103
2954 016512 060405
2955 016514 005503
2956 016516 006305
2957 016520 006103
2958 016522 110337 016174
2959 016526 062737 000060 016174
2960 016534 012704 016175
2961 016540 012737 175401 016172
2962 016546 012603
2963 016550 000137 016056
2964
2965 016554 172340
2966 016556 172240
2967 016560 172340

```

```

.SBTTL          STYPEA - 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) ; PUSH R4 ON STACK
MOV R5,-(6) ; PUSH R5 ON STACK
MOV 12(6),R5 ; R5 - OLD PS WITH PREVIOUS MODE
MOV 10(6),R4 ; R4 - ADDRESS TO BE DECODED AND TYPED
MOV 6(6),10(6) ; MOVE
MOV 4(6),6(6) ; DOWN
MOV 2(6),4(6) ; FOUR
MOV (6)+,(6) ; WORDS
MOV R3,-(6) ; PUSH R3 ON STACK
SWAB R5 ; GET THE
ROR R5 ; 2 PREVIOUS
ROR R5 ; MODE BITS
ROR R5 ; INTO POSITION
BIC #177771,R5 ; TO USE AS AN OFFSET
MOV .SATAB(5),R5 ; R5 - SPACE ADDRESS FOR MM
MOV R4,R3 ; R3 - REGISTER OFFSET
BIC #160000,R4 ; CLEAR THE MM REG SELECT BITS
SWAB R3 ; NOW MAKE
ROR R3 ; MM REG
ROR R3 ; SELECT BITS
ROR R3 ; INTO AN
ROR R3 ; OFFSET
BIC #177761,R3 ; CLEAR THE JUNK BITS
ADD R3,R5 ; ADD THE OFFSET TO THE TABLE
MOV (5),R5 ; GET THE ISAR DATA
ASL R5 ; THIS IS
ASL R5 ; TO SHIFT
ASL R5 ; THE SEGMENT
ASL R5 ; ADDRESS
CLR R3 ; INTO AN
ASL R5 ; AN 18 BIT
ROL R3 ; ADDRESS
ASL R5 ; POSITION
ROL R3 ; WITH R3 CONTAINING
ADD R4,R5 ; THE UPPER 2 BITS
ADC R3 ; AND R5 CONTAINING
ASL R5 ; THE 16 BIT ADDRESS
ROL R3 ; THEN SHIFT FOR TYPING
MOV R3,.PR+2 ; GET THE FIRST NUMBER FROM R3
ADD #0,.PR+2 ; MAKE IT INTO A NUMBER
MOV #.PR+3,R4 ; FUDGE IN THE POINTER
MOV #175401,.PR ; AND THE FLAGS (FILL & 5 BYTES)
MOV (6)+,R3 ; POP STACK INTO R3
JMP .PRL ; DECODE AND TYPE THE REST

.SATAB: 172340 ; KISARO
        172240 ; SISARO
        172340 ; KISARO - NEVER USED

```

F06

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCF.P11 16-DEC-77 15:10

MACY11 30(1046) 20-DEC-77 15:36 PAGE 71
\$TYPEA - 18 BIT ADDRESS TYPED

SEQ 0070

2968 016562 177640

177640

;UISARO

\$TRAP - TRAP HANDLER

```

2969
2970
2971
2972
2973
2974
2975
2976 016564 011646
2977 016566 162716 000002
2978 016572 017616 000000
2979 016576 062716 112204
2980 016602 013607
2981
2982 016604 015504
2983 016606 015346
2984 016610 016014
2985 016612 016024
2986 016614 021364
2987 016616 016356
2988 016620 006046
2989 016622 006070
2990 016624 017226
2991 016626 016652
2992 016630 021216
2993 016632 000316
2994 016634 014050
2995 016636 014060
2996 016640 014066
2997 016642 014100
2998 016644 017136
2999 016646 017054
3000 016650 020000

```

```

.SBTTL $TRAP - TRAP HANDLER
; THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE
; SUBROUTINE. THE CALL IS A "TRAP+N" WHERE N IS A MULTIPLE OF 2.
; THE "SET" MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO
; FOLLOW THIS MACRO.
.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
.TRP: MOV @((6)+, PC) ; GO TO PROPER SUBROUTINE

.SCOPE = TRAP+0 (104400)
.TYPE = TRAP+2 (104402)
.TYPEO = 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)

```



```

3001
3002
3003
3004
3005
3006
3007
3008 016652 010546
3009 016654 012705 016774
3010 016660 022705 017014
3011 016664 001423
3012 016666 105737 177560
3013 016672 100375
3014 016674 113715 177562
3015 016700 142715 000200
3016 016704 122715 000025
3017 016710 001006
3018 016712 104402 016716
3019 016724 000753
3020 016726 122715 000177
3021 016732 001005
3022 016734
3023 016734 104402 016740
3024 016744 000743
3025 016746 111527 000000
3026 016752 104402 016750
3027 016756 122725 000015
3028 016762 001336
3029 016764 104402 000012
3030 016770 012605
3031 016772 000002
3032
3033 016774 000020
3034 017014 000020
3035
3036 017054 005737 000042
3037 017060 001057
3038 017062 022737 000176 001026
3039 017070 001053
3040 017072 105777 161722
3041 017076 100050
3042 017100 017737 161716 017222
3043 017106 042737 177600 017222
3044 017114 122737 000007 017222
3045 017122 001036
3046 017124 104402 017130
3047 017136
3048 017136 104402 017142
3049 017152 013746 000176
3050 017156 104404
3051 017160 104402 017164
3052 017176 104420
3053 017200 012637 017222
3054 017204 005737 017224
3055 017210 001403
3056 017212 013737 017222 000176

```

```

.SBTTL          SRDLIN - TTY INPUT ROUTINE

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

RDLIN:  MOV      R5, -(6)          ; SAVE R5
1$:     MOV      #INPUT, R5       ; GET ADDRESS
2$:     CMP      #INPUT+16., R5   ; BUFFER FULL?
        BEQ      4$              ; YES - TYPE "?"
        TSTB    @#177560         ; WAIT FOR
        BPL     -4               ; A CHARACTER
        MOVB    @#177562, (5)     ; GET CHARACTER
        BICB    #200, (5)        ; GET RID OF JUNK
        CMPB    #25, (5)         ; IS IT A ↑U
        BNE     5$              ; BRANCH IF NOT
        TYPE    .+2              ; .ASCIZ "↑U"<15><12>
        BR      1$              ; START OVER
5$:     CMPB    #177, (5)        ; IS IT A RUBOUT
        BNE     3$              ; SKIP IF NOT
4$:     TYPE    .+2              ; .ASCIZ "?"<15><12>
        BR      1$              ; ZAP THE BUFFER AND LOOP
3$:     MOVB    (5), #0          ; SET UP FOR TYPING
        TYPE    3$+2            ; ECHO IT
        CMPB    #15, (5)+       ; CHECK FOR RETURN
        BNE     2$              ; LOOP IF NOT RETURN
        TYPE    12              ; TYPE A LINE FEED
        MOV     (6)+, R5         ; RESTORE R5
        RTI                    ; RETURN

INPUT:   .BLKB  16.             ; TTY INPUT AREA
ERTAB:   .BLKW  16.

.KBDIN:  TST     42              ; GOT XXDP OR ACT
        BNE    OKT              ; YES, GET OUT
        CMP    #SWREG, SWR      ; GOT SWITCH-LESS MACHINE?
        BNE    OKT              ; NO GET OUT
        TSTB  @TKS              ; HAVE A CHARACTER
        BPL   OKT              ; NO GET OUT
        MOV   @TKB, .MSG        ; STRIP OFF GARBAGE
        BIC  #177600, .MSG      ; DO WE HAVE A ↑G
        CMPB #7, .MSG           ; NO GET OUT
        BNE  OKT                ; .ASCIZ <15><12>"↑G"
        TYPE .,+2

.CNTLU:  TYPE    .+2            ; .ASCIZ <15><12>"SWR= "
        MOV     $WREG, -(6)     ; PUT SWREG ON STACK
        TYPEO  .+2            ; TYPE STACK IN OCTAL
        TYPE    .,+2            ; .ASCIZ " NEW= "
        RDOCT
        MOV     (SP)+, .MSG     ; GET NEW VALUE OFF STACK
        TST    CTN              ; DID HE TYPE <CR> OF 000000?
        BEQ    OKT              ; DONT CHANGE IF <CR>
        MOV     .MSG, SWREG     ; CHANGE VALUE OF SWREG

```

SRDLIN - TTY INPUT ROUTINE

3057 017220 000002

OKT: RTI ;ALL DONE-EXIT

3058

3059 017222 000000

.MSG: 0

3060 017224 000000

CTN: 0

3061

.SBTTL SRDOCT - OCTAL INPUT ROUTINE

3062

3063

3064

;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.

3065

3066 017226 011646

.RDOCT: MOV (6),-(6) ;MOVE THE PC

3067 017230 016666 000004 000002

MOV 4(6),2(6) ;MOVE THE PS

3068 017236 010146

MOV R1,-(6) ;PUSH R1 ON STACK

3069 017240 010246

MOV R2,-(6) ;PUSH R2 ON STACK

3070 017242 010346

MOV R3,-(6) ;PUSH R3 ON STACK

3071 017244 104422

4\$: RDLIN ;READ A LINE INTO INPUT

3072 017246 005001

CLR R1 ;INIT DATA WORD

3073 017250 005037 017224

CLR CTN ;CLEAR COUNT WORD

3074 017254 012703 016774

MOV #INPUT,R3 ;INIT POINTER

3075 017260 112302

1\$: MOVB (3)+,R2 ;GET A BYTE

3076 017262 122702 000015

CMPB #15,R2 ;WAS IT A CR?

3077 017266 001421

BEQ 2\$;GET OUT IF YES

3078 017270 122702 000060

CMPB #'0,R2 ;CHECK FOR 0 OR GREATER

3079 017274 003024

BGT 3\$;ERROR - LESS THAN 0

3080 017276 122702 000067

CMPB #'7,R2 ;CHECK FOR 7 OR LESS

3081 017302 002421

BLT 3\$;ERROR - GREATER THAN 7

3082 017304 006002

ROR R2 ;GET

3083 017306 006002

ROR R2 ;INTO

3084 017310 006002

ROR R2 ;POSITION

3085 017312 006101

ROL R1 ;FIRST BIT

3086 017314 006102

ROL R2 ;GET

3087 017316 006101

ROL R1 ;SECOND BIT

3088 017320 006102

ROL R2 ;GET

3089 017322 006101

ROL R1 ;THIRD BIT

3090 017324 005237 017224

INC CTN ;YES HE TYPED SOMETHING

3091 017330 000753

BR 1\$;LOOP

3092 017332 010166 000012

2\$: MOV R1,12(6) ;SAVE THE RESULT

3093 017336 012603

MOV (6)+,R3 ;POP STACK INTO R3

3094 017340 012602

MOV (6)+,R2 ;POP STACK INTO R2

3095 017342 012601

MOV (6)+,R1 ;POP STACK INTO R1

3096 017344 000002

RTI ;RETURN

3097

3098 017346

3\$: TYPE +2 ;ASCIZ ""<15><12>

3099 017346 104402 017352

BR 4\$;TRY AGAIN

3100 017356 000732

3101 017360 000000

OUTBUF: 0

3102

3103

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

```

3104          020000          . =      20000
3105          ;NOTE      ALL THIS CODE GETS DESTROYED WHEN PATTERN IS WRITTEN
3106
3107 020000 032737 000001 020116 .SUSWR: BIT      #BIT0,SWI
3108 020006 001037          BNE      XXX
3109 020010 013746 000006          MOV      6,-(SP)      ;SAVE 6 ON STACK
3110 020014 013746 000004          MOV      4,-(SP)      ;SAVE 4 ON STACK
3111 020020 012737 020040 000004 MOV      #1$,4        ;SET UP TRAP ADDRESS
3112 020026 022777 177777 160772 CMP      #-1,$SWR     ;TEST 177570
3113 020034 001402          BEQ      2$           ;FAKE OUT
3114 020036 000407          BR       3$           ;HARDWARE AVAILABLE
3115 020040 022626          1$: CMP      (SP)+,(SP)+ ;ADJUST STACK
3116 020042 012737 000176 001026 2$: MOV      #SWREG,SWR  ;SET UP SOFTWARE REGISTERS
3117 020050 012737 000174 001030 MOV      #DISPRÉG,DISPLAY
3118 020056 022737 000176 001026 3$: CMP      #SWREG,SWR  ;1ST TIME THRU?
3119 020064 001004          BNE      4$           ;NO CHANGE STILL 177570
3120 020066 005737 000042          TST      42          ;ANY XXDP OR ACT
3121 020072 001001          BNE      4$           ;SWR=000000
3122 020074 104440          CNTLU
3123 020076 012637 000004          4$: MOV      (SP)+,4   ;REPLACE 4 FROM STACK
3124 020102 012637 000006          MOV      (SP)+,6   ;REPLACE 6 FROM STACK
3125 020106 052737 000001 020116 XXX: BIS      #BIT0,SWI  ;SET THE BEENHEREBIT
3126 020114 000002          RTI
3127
3128 020116 000000          SWI:      0
3129
3130
3131
3132          ;ROUTINE TO SAVE ABS LOADER
3133 020120 012700 017776          LDR:  MOV      #17776,R0
3134 020124 012737 020144 000004 MOV      #2$,4        ;SET TIME OUT TRAP VECTOR
3135 020132 012737 000340 000006 MOV      #340,6
3136 020140 005720          TST      (R0)+
3137 020142 000776          BR       -2
3138 020144 022626          2$: CMP      (SP)+,(SP)+
3139 020146 012737 000006 000004 MOV      #6,4
3140 020154 005037 000006          CLR      6
3141 020160 162700 000334          SUB      #334,R0     ;POINT R0 BACK TO LOADER
3142 020164 010037 015142          MOV      R0,LDR1    ;SAVE FOR RESTORE ROUTINE
3143 020170 012702 000155          MOV      #155,R2    ;WORD COUNT
3144 020174 012703 017446          MOV      #17446,R3  ;WHERE LOADER IS TO BE STORED
3145 020200 012023 1$: MOV      (R0)+,(R3)+ ;STORE LOADER
3146 020202 005302          DEC      R2
3147 020204 001375          BNE      1$
3148 020206 000207          RTS      PC         ;RETURN
3149
3150
3151          ; -A- PORT SIZE
3152
3153 020210 052737 020000 001122 SIZZAP: BIS      #BIT13,FLAG2 ;SET MAPPING BIT
3154 020216 042737 000500 001126 BIC      #500, FLAG  ;CLEAR FLAG BEFORE READ
3155 020224 042737 000460 001122 BIC      #460, FLAG2 ;CLEAR FLAG2 BEFORE READ
3156 020232 042737 003400 001172 BIC      #3400, CMD  ;CLEAR CMD BEFORE READ
3157 020240 004737 001624          JSR      PC,DRVENO  ;FIND DRIVE
3158 020244 012737 000002 001224 MOV      #2,WORK1   ;START WITH ONE 4K BUFFER
3159 020252 012737 000001 001070 MOV      #1,STAMEM  ;FIRST 4K BANK

```

K06

CZRSCF RH11-RS03-RS03/LA-RS04
CZRSCF.P11 16-DEC-77 15:10

DATA AND RELIABILITY TEST
SRDOCT - OCTAL INPUT ROUTINE

MACY11 30(1046) 20-DEC-77 15:36 PAGE 76

SEQ 0075

3160	020260	012737	057476	001140		MOV	#57476, BUF	; GET STARTING ADDR. 5K
3161	020266	012737	000001	001130		MOV	#1, WRDCT	; LOAD WC
3162	020274	005037	001134			CLR	DMA	; LOAD DA
3163	020300	012777	000040	160526		MOV	#40, @RSCS2	; CLEAR ALL RS REG
3164	020306	013777	001160	160520		MOV	UNNUM, @RSCS2	; GET DRIVE #
3165	020314	012737	000071	001172		MOV	#71, CMD	; DO A READ
3166	020322	104416			4\$:	DKCMD		; NOW
3167	020324	105777	160502		1\$:	TSTB	@RSCS1	; DONE YET?
3168	020330	100375				BPL	1\$; NO
3169	020332	032777	004000	160474		BIT	#4000, @RSCS2	; DID NEM SET?
3170	020340	001012				BNE	SIZ1	; YES
3171	020342	005777	160464			TST	@RSCS1	; ANY ERRORS?
3172	020346	100005				BPL	3\$; NO
3173	020350	012737	000006	001106		MOV	#6, SIZEAP	; GET SIZE OF BUFFER
3174	020356	000137	020572			JMP	@#SIZERR	; FOR USER IF HE WISHES IT
3175	020362	104424			3\$:	UPDAT		; GET NEXT 4K BANK
3176	020364	000756				BR	4\$; TEST BANK
3177	020366	005337	001224		SIZ1:	DEC	WORK1	; DEC SIZE OF BUFFER
3178	020372	013737	001224	001106		MOV	WORK1, SIZEAP	; LOAD SIZE OF A BUFFER
3179	020400	104402	020404			TYPE	+2	; .ASCIZ <15><12> "PORT -A- DATA BUFFER 4K TO "
3180	020446	004737	021156			JSR	PC, SIZPR	

; -B- PORT SIZE

3181										
3182										
3183	020452	012737	000001	001224	SIZZBP:	MOV	#1,WORK1		: START WITH ONE 4K BUFFER	
3184	020460	042737	000500	001126		BIC	#500, FLAG		: CLEAR FLAG BEFORE READ	
3185	020466	042737	000460	001122		BIC	#460, FLAG2		: CLEAR FLAG2 BEFORE READ	
3186	020474	042737	003400	001172		BIC	#3400, CMD		: CLEAR CMD BEFORE READ	
3187	020502	012737	037476	001140		MOV	#37476, BUF		: GET STARTING ADDR. 4K	
3188	020510	012737	000001	001130		MOV	#1,WRDCT		: LOAD WC	
3189	020516	005037	001134			CLR	DMA		: LOAD DA	
3190	020522	012777	000040	160304		MOV	#40,DRSCS2		: CLEAR ALL RS REG	
3191	020530	013777	001160	160276		MOV	UNNUM,DRSCS2		: GET DRIVE #	
3192	020536	012737	002071	001172		MOV	#2071,CMD		: DO A READ	
3193	020544	104416			SIZ2:	DKCMD			: NOW	
3194	020546	105777	160260		1\$:	TSTB	DRSCS1		: DONE YET?	
3195	020552	100375				BPL	1\$: NO	
3196	020554	032777	004000	160252		BIT	#4000,DRSCS2		: DID NEM SET?	
3197	020562	001067				BNE	SIZB1		: YES	
3198	020564	005777	160242			TST	DRSCS1		: ANY ERRORS?	
3199	020570	100047				BPL	SIZ3		: NO	
3200	020572				SIZERR:					
3201	020572	104402	020576			TYPE	+2		: .ASCIZ <15><12>"WILL NOT CONTINUE TO SIZE MEMORY BECAUS	
3202	020662	012737	000006	001110		MOV	#6,SIZEBP		: GIVE PROGRAM A BUFFER	
3203	020670	104060				HLT	!BA!DS		: YOU CAN ENTER CONVERSATION MODE	
3204	020672	052737	000001	001122		BIS	#BIT0,FLAG2		: BEEN HERE BEFORE FLAG	
3205	020700	042737	020000	001122		BIC	#BIT13,FLAG2		: CLEAR MAPPING FLAG	
3206	020706	000000				HALT			: OR GO TO DZRSB	
3207	020710	032737	000400	001122	SIZ3:	BIT	#BIT8,FLAG2		: FOUND MEMORY YET?	
3208	020716	001006				BNE	SIZB3		: YES	
3209	020720	052737	000400	001122		BIS	#BIT8,FLAG2		: SET FOUND MEM FLAG	
3210	020726	013737	001224	001074		MOV	WORK1,STBCOM		: GET 1ST BANK	

M06

3211	020734	104424			SIZB3:	UPDAT				;GET NEXT 4K BANK
3212	020736	000702				BR	SIZ2			;TEST BANK
3213	020740	000404				BR	SIZB2			
3214	020742	032737	000400	001122	SIZB1:	BIT	#BIT8,FLAG2			;FOUND MEMORY?
3215	020750	001771				BEQ	SIZB3			;NO
3216	020752	005337	001224		SIZB2:	DEC	WORK1			;DEC SIZE OF BUFFER
3217	020756	013737	001224	001110		MOV	WORK1,SIZEBP			;LOAD SIZE OF B BUFFER
3218	020764	032737	000400	001122		BIT	#BIT8,FLAG2			;FOUND MEMORY?
3219	020772	001017				BNE	6\$;YES
3220	020774	104402	021000			TYPE	+2			;ASCIZ <15><12>"NO MEMORY ON -B- PORT"
3221	021030	000442				BR	1\$			
3222	021032				6\$:					
3223	021032	104402	021036			TYPE	+2			;ASCIZ <15><12> "PORT -B- DATA BUFFER"
3224	021066	013737	001074	001224		MOV	STBCOM,WORK1			
3225	021074	005001				CLR	R1			
3226	021076	005002				CLR	R2			
3227	021100	004737	021164			JSR	PC, SIZP			
3228	021104	104402	021110			TYPE	+2			;ASCIZ " TO "
3229	021116	013737	001110	001224		MOV	SIZEBP,WORK1			
3230	021124	004737	021156			JSR	PC, SIZPR			
3231	021130	052737	000100	001126		BIS	#BIT6,FLAG			;SET MULTI PORT FLAG
3232	021136	042737	020000	001122	1\$:	BIC	#BIT13,FLAG2			;CLEAR MAPPING FLAG
3233	021144	052737	000002	001122		BIS	#BIT1,FLAG2			;SET BEEN HERE FLAG
3234	021152	000137	001474			JMP	CALM			;CAL BUFFER AND WC
3235										
3236	021156	005001			SIZPR:	CLR	R1			;INIT SETUP
3237	021160	012702	000004			MOV	#4,R2			
3238	021164	062701	000001		SIZP:	ADD	#1,R1			;SETUP FOR BANK NO
3239	021170	062702	000004			ADD	#4,R2			;SETUP FOR SIZE FO MEMORY
3240	021174	020137	001224			CMP	R1,WORK1			;IS THIS THE RIGHT SIZE?
3241	021200	001371				BNE	SIZP			;NO
3242	021202	010246				MOV	R2,-(6)			;PUT R2 ON STACK
3243	021204	104410				TYPED				;TYPE STACK IN DECIMAL
3244	021206	104402	021212			TYPE	+2			;ASCIZ "K"
3245	021214	000207				RTS	PC			;RETURN
3246										
3247										
3248										
3249	021216	005237	001224							
3250	021222	062737	020000	001140	.UPDAT:	INC	WORK1			;INC BANK #
3251	021230	005737	001212			ADD	#20000,BUF			;UPDATE BY 4K
3252	021234	001437				TST	MMAVA			;MEMORY MANAGEMENT AVAILABLE?
3253	021236	022737	000037	001224		BEQ	3\$;NO
3254	021244	001440				CMP	#37, WORK1			;BANK 37?
3255	021246	022737	000027	001224		BEQ	1\$;YES-STOP SIZING
3256	021254	002004				CMP	#27, WORK1			;BANK 30?
3257	021256	052737	001400	001172		BGE	10\$;NO
3258	021264	000002				BIS	#1400, CMD			;YES-ENABLE A16 AND A17
3259	021266	022737	000017	001224	10\$:	RTI				;CONTINUE
3260	021274	002007				CMP	#17, WORK1			;BANK 20?
3261	021276	052737	001000	001172		BGE	11\$;NO
3262	021304	042737	000400	001172		BIS	#1000, CMD			;YES-ENABLE A17
3263	021312	000002				BIC	#400, CMD			;DISABLE A16
3264	021314	022737	000007	001224	11\$:	RTI				;CONTINUE
3265	021322	002010				CMP	#7, WORK1			;BANK 10?
3266	021324	052737	000400	001172		BGE	2\$;NO
						BIS	#400, CMD			;YES-ENABLE A16

3267	021332	000002				RTI			;CONTINUE
3268	021334	022737	177476	001140	3\$:	CMP	#177476,BUF		;EXCEEDED MEM YET?
3269	021342	001401				BEQ	1\$;YES
3270	021344	000002			2\$:	RTI			;CONTINUE
3271	021346	062716	000002		1\$:	ADD	#2,(6)		;UPDATE RETURN PC BY 2
3272	021352	042737	003400	001172		BIC	#3400, CMD		;CLEAR CMD AFTER FINISHING
3273	021360	000002				RTI			
3274									
3275	021362	021364				.TYPED			;TYPED = TRAP+46 (104446)
3276						.SBTTL	\$TYPED - CONVERT		;BINARY TO DECIMAL AND TYPE ROUTINE
3277									
3278	021364	012737	100040	021612	.TYPED:	MOV	#100040,.DSIGN		;SET BLANK SWITCH AND SIGN
3279	021372	010046				MOV	R0,-(6)		;PUSH R0 ON STACK
3280	021374	010146				MOV	R1,-(6)		;PUSH R1 ON STACK
3281	021376	010246				MOV	R2,-(6)		;PUSH R2 ON STACK
3282	021400	010346				MOV	R3,-(6)		;PUSH R3 ON STACK
3283	021402	010546				MOV	R5,-(6)		;PUSH R5 ON STACK
3284	021404	012737	100040	021612		MOV	#100040,.DSIGN		;SET BLANK SWITCH AND SIGN
3285	021412	016605	000016			MOV	16(6),R5		;GET DATA TO BE TYPED
3286	021416	100004				BPL	1\$;BR IF INPUT IS POS.
3287	021420	005405				NEG	R5		;MAKE THE BINARY NUMBER POS.
3288	021422	112737	000055	021612		MOVB	#',.DSIGN		;MAKE THE ASCII NUMBER NEG.
3289	021430	005000			1\$:	CLR	R0		;ZERO THE CONSTANTS INDEX
3290	021432	012703	021602			MOV	#,DBLK,R3		;SETUP THE OUTPUT POINTER
3291	021436	112723	000040			MOVB	#',(R3)+		;SET THE FIRST CHARACTER TO A BLANK
3292	021442	005002			2\$:	CLR	R2		;CLEAR THE BCD NUMBER
3293	021444	016001	021572			MOV	.DTBL(R0),R1		;GET THE CONSTANT
3294	021450	160105			3\$:	SUB	R1,R5		;FORM THIS BCD DIGIT
3295	021452	002402				BLT	4\$;BR IF DONE
3296	021454	005202				INC	R2		;INCREASE THE BCD DIGIT BY 1
3297	021456	000774				BR	3\$		
3298	021460	060105			4\$:	ADD	R1,R5		;ADD BACK THE CONSTANT
3299	021462	005702				TST	R2		;CHECK IF BCD DIGIT=0
3300	021464	001003				BNE	5\$;FALL THROUGH IF 0
3301	021466	105737	021613			TSTB	.DSIGN+1		;STILL DOING LEADING 0'S?
3302	021472	100410				BMI	7\$;BR IF YES
3303	021474	106337	021613		5\$:	ASLB	.DSIGN+1		;MSD?
3304	021500	103003				BCC	6\$;BR IF NO
3305	021502	113763	021612	177777		MOVB	.DSIGN,-1(R3)		;YES--SET THE SIGN
3306	021510	052702	000060		6\$:	BIS	#'0,R2		;MAKE THE BCD DIGIT ASCII
3307	021514	052702	000040		7\$:	BIS	#',R2		;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3308	021520	110223				MOVB	R2,(R3)+		;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3309	021522	005720				TST	(R0)+		;JUST INCREMENTING
3310	021524	020027	000010			CMP	R0,#10		;CHECK THE TABLE INDEX
3311	021530	002744				BLT	2\$;GO DO THE NEXT DIGIT
3312	021532	003002				BGT	8\$;GO TO EXIT
3313	021534	010502				MOV	R5,R2		;GET THE LSD
3314	021536	000764				BR	6\$;GO CHANGE TO ASCII
3315	021540	105013			8\$:	CLRB	(R3)		;SET THE TERMINATOR
3316	021542	012605				MOV	(6)+,R5		;POP STACK INTO R5
3317	021544	012603				MOV	(6)+,R3		;POP STACK INTO R3
3318	021546	012602				MOV	(6)+,R2		;POP STACK INTO R2
3319	021550	012601				MOV	(6)+,R1		;POP STACK INTO R1
3320	021552	012600				MOV	(6)+,R0		;POP STACK INTO R0
3321	021554	016666	000002	000004		MOV	2(6),4(6)		;FUDGE DATA
3322	021562	012616				MOV	(6)+,(6)		;OFF STACK

3323 021564 104402 021602 TYPE ,.DBLK ;NOW TYPE THE NUMBER
3324 021570 000002 RTI ;RETURN
3325
3326 021572 023420 001750 000144 .DTBL: 10000.,1000.,100.,10.
3327 021600 000012
3328 021602 000004 .DBLK: .BLKW 4
3329 021612 000000 .DSIGN: 0
3330
3331 000001 .END

ADAM	010624	2018	2023*	2032														
ADDR	014106	2539	2546	2551*														
ADERR	003554	1233	1238*															
ADHGT	003512	1222	1227*															
ADRO	004660	1386	1403*															
ADR1	004674	1402	1405	1407*														
ADTL	003246	619	1169*															
ADTST	003154	611	963	1151	1154*	1742	2533											
ADT1	003254	1162	1179*															
ADT1A	003256	1184*	1255															
AKH	007300	1799*	1818															
APOINT	011276	1268	2104*															
AS =	000100	810*	1644	2588														
ASKWC	002500	1062	1080*															
A0B1	001114	830*	932*	1043*	1092	1100	1266	1420*	1625	1709	1806	1966	2447*	2506*				
A1	002330	615	1055*															
BA =	000020	808*	1211	1352	1511	2593	3203											
BEGIN	001234	910*	1977	1996	2284	2662												
BELL =	000007	697*																
BIT0 =	000001	706*	919	925	943	948	952	2643	3107	3125	3204							
BIT1 =	000002	707*	956	969	1023	1034	1078	1416	3233									
BIT10 =	002000	716*	958	959	1055	1161	1418	1627	1658	2258	2285							
BIT11 =	004000	717*	1061	1154	1578	1655	2698											
BIT12 =	010000	718*	1146	1368	1533	2094	2441	2635										
BIT13 =	020000	719*	1152	1310	1599	2563	3153	3205	3232									
BIT14 =	040000	595	720*	1001	1140	1280	1681	1683										
BIT15 =	100000	721*	1131	1169	1253	1425	1565	1974	2350	2358	2407	2417	2663	2710				
BIT2 =	000004	708*	933	983	1008	1015	1157	1160	1184	1382	1435	1601	1752	1878				
BIT3 =	000010	2092	2440	2565														
BIT4 =	000020	709*	1704	1984	2023	2101	2464	2699										
BIT5 =	000040	710*	1620	1707	2087	2223	2226											
BIT6 =	000100	711*	980	1006	1107	1617	1667	1747	2220	2222	2507							
		712*	1052	1264	1322	1387	1469	1497	1540	1623	1804	2027	2141	2151				
BIT7 =	000200	2171	3231															
BIT8 =	000400	713*	816	1004	2194	2370	2429											
		714*	1158	1189	1252	1335	1438	1564	1613	1622	1702	1800	3207	3209				
BIT9 =	001000	3214	3218															
		715*	1209	1223	1286	1317	1324	1385	1467	1495	1537	1542	1587	1590				
		1619	1634	1654	1705	1778	2036	2331	2347	2390	2404	2412	2418	2428				
BLOCK	001154	2456	2640															
		884*	1771	1775*	1838*	1840*	1843*	1845	1847	1848*	1849	1862*	2054*	2068*				
		2080																
BLSZ	007504	1770	1836*															
BPORT	011344	1270	2114*															
BUF	001140	878*	1195*	1274*	1313*	1461*	1609	2448*	3160*	3187*	3250*	3268						
BUFEXI	007270	1749	1757	1763	1767	1774	1797*											
BUFINX	007354	1796	1810*															
CALM	001474	944	946*	3234														
CALM1	001500	947*																
CHCKDV	002150	1018	1023*															
CHKADT	003600	1237	1243	1248*														
CHKDAT	012570	2342*	2354															
CHKDSK	013034	2400*	2411															
CLEAR	011230	1366	1528	2088	2090	2092*												
CLIND	014054	2538*	2542															
CLRDV =	104426	1192	1262	1445	1577	1589	2323	2341	2392	2396	2993*							

SI23	020710	3199	3207#														
SLH	004124	1281	1307#														
SLH2	004126	1308#	1356	1363	1375												
SLH2A	004514	1365	1375#														
SR0	= 177572	652#	937*	1443*	2250*												
SR1	= 177574	653#															
SR2	= 177576	654#															
SR3	= 172516	655#															
STABUF	000653	755#	1045	1049													
STAMEM	001070	820#	1047*	1708	2132	2160	3159*										
STATUS	001064	815#	1272*	1449*													
STBCOM	001074	822#	1051*	1711	2143	2153	2175	3210*	3224								
STMM2	011762	2109	2209#														
STTEST	001736	981	984#	1020													
SUSWR	= 104444	921	3000#														
SWI	020116	909*	919	3107	3125*	3128#											
SWITCH	001174	892#	1987*	2004	2008*	2010*											
SWR	001026	784#	933	956	959	980	1006	1158	1322	1324	1335	1380	1387	1416			
		1469	1497	1533	1540	1542	1655	1974	2027	2036	2087	2194	2464	2507			
		2563	2640	2643	2710	2763	2774	2804	2814	2817	3038	3112	3116*	3118			
SWRDCT	001144	880#	950	955	1054	1098*	1099	1190	1194*	1251*	1263	1802	1928*	1929			
		2168*	2170*	2445*	2446	3056*	3116	3118									
SWREG	000176	599#	3038	3049													
SW0	= 000001	590#															
SW1	= 000002	589#															
SW10	= 002000	580#	2528	2802													
SW11	= 004000	579#	2774														
SW12	= 010000	578#	2528														
SW13	= 020000	577#	2804														
SW14	= 040000	576#	2763														
SW15	= 100000	575#															
SW2	= 000004	588#															
SW3	= 000010	587#															
SW4	= 000020	586#															
SW5	= 000040	585#															
SW6	= 000100	584#															
SW7	= 000200	583#															
SW8	= 000400	582#	2762														
SW9	= 001000	581#	2817														
TDMA	001142	879#	1121*	1669													
TIMES	015656	2778	2793#														
TIMUP	006740	1735#	2905														
TKB	001022	782#	3042														
TKS	001020	781#	3040														
TPB	001024	783#	2736*	2741*													
TPS	001016	780#	2737	2742													
TRACK	001132	875#															
TRUERR	006364	1641	1643	1654#													
TRY	001644	971#	975														
TRYNX	002134	985	999	1005	1017#	1580											
TSTNG	000706	761#	982														
TST1	003254	1183#															
TST2	003640	1260#															
TST3	005004	1434#															
TST4	006012	1575#															
TYPE	= 104402	982	1010	1014	1040	1045	1049	1058	1063	1081	1086	1104	1108	1123			

NO7

CZRSCF RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
CZRSCF.P11 16-DEC-77 15:10

MACY11 30(1046) 20-DEC-77 15:36 PAGE 94
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0091

. ABS. 021614 000

ERRORS DETECTED: 0

CZRSCF.BIN,CZRSCF.LST/CRF/SOL/NL:TOC=CZRSCF.SML,CZRSCF.P11

RUN-TIME: 7 11 1 SECONDS
RUN-TIME RATIO: 141/20=6.9
CORE USED: 22K (43 PAGES)

