

RP04
RP05, RP06

RP04/5/6 DSKLS 1
CZRJGEO

AH-9210E-MC
FICHE 1 OF 2

APR 1982
COPYRIGHT © 76-82
MADE IN USA



A large grid of approximately 15 columns and 20 rows of small, dense data tables. Each cell in the grid contains a small table with multiple columns and rows of text, likely representing individual data points or small reports. The text is too small to read clearly but appears to be organized in a structured format.

RP04
RP05, RP06

RP04/5/6 DSKLS 1
CZRJGEO

AH-9210E-MC
FICHE 2 OF 2

APR 1982
COPYRIGHT © 76-82
MADE IN USA



[Faded microfiche data columns]



.REM @

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

IDENTIFICATION

PRODUCT CODE: AC-9208E-MC
PRODUCT NAME: CZRJGEO RP04/5/6 DISKLESS TEST, PT 1
PRODUCT DATE: NOVEMBER 1981
MAINTAINER: CX DIAGNOSTIC GROUP
AUTHOR: PETE BLACKSTONE
REVISED BY: MIKE LEAVITT

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

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

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

COPYRIGHT (C) 1976,1979,1982 DIGITAL EQUIPMENT CORPORATION

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

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
- 3. LOADING PROCEDURE
 - 3.1 METHOD
- 4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS OR ADDRESSES
 - 4.3 PROGRAM AND/OR OPERATOR ACTION
- 5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUB-ROUTINE ABSTRACTS
- 6. ERRORS
 - 6.1 'FATAL' ERRORS
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 OPERATOR SELECTABLE SCOPE LOOPS
- 9.0 PROGRAM DESCRIPTION

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

1.0 ABSTRACT

THE DIAGNOSTIC IS USED TO TEST RP04/5/6 DEVICE CONTROL LOGIC CONNECTED TO EITHER AN RH11 OR RH70 DISK DRIVE CONTROLLER

THIS DIAGNOSTIC TESTS THE RH11 AND DCL OF AN RJP04/5/6 SUBSYSTEM. IT DOES NOT USE THE DISK SURFACE OR ANY SIGNALS FROM THE MDLI. IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDLI OR BE APPROPRIATELY TERMINATED. IF THE DISK IS POWERED UP, IT IS REQUIRED TO GET THE DISK TO THE 'HEADS UNLOADED' POSITION. AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF THIS DIAGNOSTIC IT CAN BE ASSERTED THAT, 'THAT PART OF THE DCL THAT HANDLES DATA OR DATA ASSOCIATED LOGIC IS WORKING PROPERLY'. THIS IMPLIES THAT, THE PART OF THE LOGIC WHICH HANDLES MECHANICAL COMMANDS OR ITS ASSOCIATED LOGIC IS NOT TESTED IN THIS DIAGNOSTIC. ALL DATA COMMANDS USE THE MAINTENANCE REGISTER IN THE WRAPAROUND MODE.

THE DIAGNOSTIC DOES NOT DO ANY TESTING OF THE RH70 CONTROLLER WHEN IT IS USED ON AN RWPO4/5/6 SYSTEM TO TEST RP04/5/6 DISK DRIVES CONNECTED TO THAT TYPE OF CONTROLLER. IT IS ASSUMED THAT THE RH70 SPECIFIC CONTROLLER DIAGNOSTIC HAVE BEEN SUCCESSFULLY RUN TO COMPLETION BEFORE THIS DIAGNOSTIC IS RUN.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH4 CONSOLE TELETYPE, AND A RP04/5/6 DISK SYSTEM. THE RP04/5/6 DISK SYSTEM WILL CONSIST OF AN RH11/RH70 CONTROLLER, AND DISK CONTROL LOGIC (DCL), THE CABLE FROM THE DCL CAN BE CONNECTED TO THE MDLI, BUT IF NOT THAT CABLE MUST BE PROPERLY TERMINATED.

2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

THIS CAN BE THE FIRST PROGRAM RUN ON AN RJP04/5/6 SYSTEM BUT THE CONTROLLER DIAGNOSTICS MUST BE RUN FIRST IN THE CASE OF AN RWPO4/5/6 SYSTEM.

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

SWITCH 12 MUST BE SET WHEN THIS PROGRAM IS TO BE RUN USING AN RH70 CONTROLLER. IT CAN BE SET AT THE FRON PAEL, OR IN THE SOFTWARE SWITCH REGISTER IF THE OPERATOR SO DESIRES. SE PARAGRAPH 5.1 FOR A DESCRIPTION OF SOFTWARE SWITCH REGISTER OPERATION.

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
101
102
103
104
105
106
107
108
109
110
111
112
113
114

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

START AT ADDRESS 200---FOR NORMAL RUN
START AT ADDRESS 204---TO SELECT NON-DEFAULT ADDRESSES
START AT ADDRESS 210---FOR UNIT SELECTION

200 START

ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS ALL THE RP04/5/6S ON THE SYSTEM WILL BE TESTED ONE AT A TIME BEFORE 'END PASS' IS PRINTED OUT. TESTING WILL START WITH THE LOWEST UNIT NUMBER DRIVE THAT IS POWERED UP (THAT IS THE LOWEST UNIT NUMBER RHAS REGISTER THAT RESPONDS) THEN GO ON TO THE NEXT HIGHER UNIT NUMBER THAT IS POWERED UP.

204 RESTART

SAME AS START 200 WITH THE FOLLOWING EXCEPTION: THE PROGRAM WILL INTERROGATE THE OPERATOR FOR A NON-STANDARD C.S.R. AND VECTOR ADDRESS BEFORE STARTING. ONCE THE QUESTIONS HAVE BEEN CORRECTLY ANSWERED, AND IT IS ALSO NECESSARY TO SELECT A PARTICULAR UNIT FOR TEST (TYPICAL PROGRAM EXECUTION FROM ADDRESS 210), THE PROCESSOR MAY BE HALTED AND RESTARTED FROM ADDRESS 210. THE NEW PARAMETERS WILL NOT BE CHANGED UNLESS THE PROGRAM IS AGAIN RESTARTED FROM ADDRESS 204. IF ALL UNITS ARE TO BE CHECKED, THE PROCESSOR NEED NOT BE TOUCHED. THE PROGRAM WILL AUTOMATICALLY RESTART AT 200 AFTER RECEIVING THE NEW DEVICE PARAMETERS.

210 START

ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS THE CONSOLE TELETYPE WILL ASK FOR THE UNIT NUMBER TO BE TESTED. THEN ONLY THAT UNIT WILL BE TESTED FOR EACH PASS OF THE PROGRAM.

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD THE PROGRAM INTO MEMORY.
2. SET STARTING ADDRESS ON THE SWITCH REGISTER
3. PRESS 'LOAD ADDRESS'.
4. SET 'OPERATIONAL SWITCH SETTINGS' (SEE SECTION 5.1) WORST CASE IS ALL SWITCHES DOWN.
5. PRESS 'START'.
6. FOR THE FIRST PASS EACH TEST WILL BE EXECUTED ONCE ON THE DRIVES PRESENT OR DRIVE SELECTED BEFORE 'END PASS' IS PRINTED. THE FIRST PASS WILL REQUIRE OPERATOR INTERVENTION IF THE PROGRAM IS NOT RUN UNDER AN 'ACT-11' MONITOR. THE SECOND AND SUBSEQUENT PASSES WILL EXECUTE

115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171

EACH TEST FOUR TIMES ON EACH DRIVES PRESENT OR DRIVE
SELECTED BEFORE 'END PASS' IS PRINTED. THE SECOND
AND SUBSEQUENT PASSES DO NOT NEED ANY OPERATOR INTERVENTION.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I. E.
AN 11/34) IT WILL DETERMINE THAT A HARDWARE SWITCH REGISTER
IS NOT PRESENT, AND WILL USE A 'SOFTWARE' SWITCH REGISTER.
THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED
THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A
'CONTROL G'. THE PROGRAM WILL RECOGNIZE A 'CONTROL G' AT ANY
TIME EXCEPT WHEN IT IS AT A HIGHER PRIORITY PROCESSING A RP04/5/6
INTERRUPT. THE 'SOFTWARE' SWITCH VALUEA ARE ENTERED AS AN
OCTAL NUMBER IN RESPONSE TO A PROMPT FROM THE SWITCH ENTRY
ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH
REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT
REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO
CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE'
SWITCH REGISTER MAY ALSO BE USED. IF THE PROGRAM FINDS ALL
16 SWITCHES IN THE 'UP' POSITION WHEN IT IS STARTED, ALL
SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER
AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

SWITCH DEFINITIONS ARE GIVEN IN SECTION 9 'OPERATIONAL
SWITCH SETTINGS' HOWEVER THE DETAIL DESCRIPTION ARE GIVEN
HERE.

SWITCH 15 - HALT ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR
THEN THE APPROPRIATE INFORMATION WILL BE PRINTED OUT
AND THEN THE PROGRAM WILL HALT. AFTER THIS HALT, PRESSING
'CONTINUE' WILL CONTINUE WITH THE PROGRAM TILL THE NEXT
ERROR IS FOUND WHEN THE SAME THING WILL HAPPEN.

SWITCH 14 - LOOP ON TEST
WHEN THIS SWITCH IS SET THE PROGRAM WILL BEGIN TO LOOP
ON THE CURRENT TEST BEING EXECUTED. FOR EXAMPLE IF THIS
SWITCH IS SET WHEN THE PROGRAM IS IN TEST 10 THEN THE
PROGRAM WILL KEEP EXECUTING ALL OF TEST 10 REPEATEDLY.
ONE WAY TO BE SURE THAT THE PROGRAM IS IN THE EXPECTED
TEST IS TO SET THIS SWITCH DURING AN ERROR PRINTOUT OR
DURING A PROGRAM HALT.

SWITCH 13 - INHIBIT ERROR TYPEOUTS
WHEN THIS SWITCH IS SET FURTHER ERROR PRINTOUTS WILL
CEASE, HOWEVER OPERATOR INSTRUCTIONS SUCH AS 'STOP DRIVE X'

172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228

WILL CONTINUE. AT THE END OF PASS 'TOTAL NUMBER OF ERRORS ON THIS PASS ON DRIVE X' WILL BE TRUE, THAT IS, ALTHOUGH PRINTOUTS WERE INHIBITED IF THAT PASS FOUND 6 ERRORS, IT WILL SAY SO.

SWITCH 12 - RH70 CONTROFLLER SELECT
THIS SWICH MUST BE SET AT THE START OF THE PROGRAM WHEN THE DISK DRIVES TO BE TESTED ARE CONNECTED TO AN RH70 CONTROLLER. IT MUST NOT BE SET WHEN DISK DRIVES TO BE TESTED ARE CONNECTED TO AN RH11 CONTROLLER.

SWITCH 11 - INHIBIT ITERATIONS
WHEN THIS SWITCH IS SET THE PROGRAM ON SECOND PASS WILL NOT REPEAT EACH TEST FOUR TIMES BUT WILL DO EACH TEST ONCE ONLY.

SWITCH 10 - BELL ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR THE 'BELL' OR 'ALARM' WILL BE SOUNDED. THIS SWITCH IS USEFUL WHEN SWITCH 11 IS SET YET INFORMATION IS NEEDED WHEN ANY ERROR IS DETECTED. TAKE THE EXAMPLE OF A PROGRAM LOOPING ON A TEST WITH SWITCH 11 SET TO HELP SCOPING. THEN IF THIS SWITCH IS SET AND THE BELL OR ALARM SOUNDS IT MEANS THAT THE ERROR IS PRESENT BUT IF THE BELL OR ALARM STOPS IT MEANS THAT THE ERROR IS NOT PRESENT.

SWITCH 9 - LOOP ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR THEN GENERALLY THE PROGRAM WILL LOOP BACK TO THE LAST EXECUTED 'SCOPE' STATEMENT. IF ON THE SECOND TIME THROUGH AN ERROR IS FOUND IT WILL AGAIN LOOP BACK TO THAT 'SCOPE' STATEMENT. THIS LOOPING WILL CONTINUE AS LONG AS THE ERROR IS PRESENT AND THIS SWITCH IS SET. HOWEVER IF THE ERROR IS NOT PRESENT AT ANY TIME THEN IT WILL CONTINUE NORMALLY WITH THE PROGRAM. EACH TIME THE ERROR IS ENCOUNTERED PRINTOUT WILL TAKE PLACE UNLESS SWITCH 11 IS ALSO SET. DURING BEGUG, USING A SCOPE, IT IS RECOMMENDED THAT SWITCH 11 IS ALSO SET.

NOTE: ALSO SEE SECTION 8.3

SWITCH 8 - LOOP ON TEST IN SWR <7:0>
THIS IS A SPECIAL SWITCH. WHEN SET SWITCHES 0 THRU 7 HAVE ONE MEANING AND WHEN RESET SWITCHES 0 THRU 7 HAVE ANOTHER MEANING. THIS MEANS THAT ANY SETTING OF SWITCH 0 THRU 7 MUST BE DONE WITH SWITCH 8 IN THE APPROPRIATE POSITION. WHEN THIS SWITCH IS SET THEN SWITCHES 0 THRU 7 GIVE THE TEST NUMBER TO BE LOOPED ON. FOR EXAMPLE WITH SWITCH 8 SET AND SWITCH 3 SET THE PROGRAM WILL LOOP ON TEST 10. HOWEVER THIS SETTING MUST BE DONE AT THE BEGINNING OF THE PROGRAM THEN ALL THE TESTS FROM 1 TO 10 WILL BE EXECUTED AND THEN TEST 10 WILL BE REPEATED OVER AND OVER AGAIN. WHEN THIS SWITCH IS NOT SET THEN SWITCHES

229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
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
283
284
285

0 THRU 7 HAVE THE MEAING ITS NAME INDICATES. FOR EXAMPLE SWITCH 7 IS "STOP FURTHER COMPARES: THAT IS IF SWITCH 8 IS NOT SET AND SWITCH 7 IS SET THEN WHEN A DATA ERROR IS DETECTED NO FURTHER COMPARES WILL BE DONE. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE PRINTOUT FOR THE FIRST FEW WORDS SETTING SWITCH 7 ONLY WILL STOP FURTHER PRINTOUTS OF THIS ERROR AND GO ON WITH THE TEST RATHER THAN PRINT ALL THE 256 WORDS. HOWEVER IF THIS WAS DONE WITH SWITCH 11 THEN THE NEXT ERROR THAT THE PROGRAM DETECTS IN A SUBSEQUENT TEST WILL ALSO BE LOST. BUT WITH SWITCH 7, ONLY THIS GROUP OF DATA ERRORS ARE NOT PRINTED OUT. ANOTHER EXAMPLE OF SWITCH 8 BEING LOW IS WITH SWITCH 6, WHICH IS "ECC TEST-COMPARE END RESULT ONLY". THAT IS IF SWITCH 8 IS NOT SET AND SWITCH 6 IS SET THEN ON ECC TESTS (TEST 120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE POSITION REGISTER AND PATTERN REGISTER AFTER EVERY CLOCK, COMPARES WILL ONLY BE DONE AT THE END OF ALL THE CLOCKS.

NOTE: ALSO SEE SECTION 8.3

SWITCH 7 - STOP FURTHER COMPARES IF SW08 IS LOW. IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN THE PROGRAM WILL DO AS THE NAME INDICATES. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE ERROR PRINTOUTS FOR THE FIRST FEW WORDS THEN SETTING SWITCH 7 WITH SWITCH 8 NOT SET WILL STOP THE PRINTOUT OF ALL 256 WORDS BUT WILL NOT STOP THE PRINTOUT OF ANOTHER ERROR IN ANY SUBSEQUENT TEST. IT IS EXPECTED THAT SWITCH 7 AFTER BEING SET FOR A WHILE TO STOP PRINTING ALL THE 256 WORDS WILL BE RESET AGAIN TO ENABLE THE PRINTING OF OTHER DATA ERRORS.

SWITCH 6 - ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN ON ECC TESTS (TEST 120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE POSITION AND PATTERN REGISTERS AFTER EVERY CLOCK, COMPARES WILL BE DONE ONLY AT THE END OF ALL THE CLOCKS.

5.2 SUB-ROUTINE ABSTRACTS

SEE SECTION 9 "SUBROUTINES"

6.0 ERRORS

ERROR PRINTOUTS CONTAIN THE ERROR ADDRESS AND OTHER PERTINENT INFORMATION CONCERNING THE PARTICULAR FAILURE. THIS INFORMATION MAY BE THE CONTENTS OF RELEVANT RP04/5/6

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
331
332
333
334
335
336
337
338
339
340
341
342

REGISTERS OR GOOD/RECEIVED DATA. IF THE ERROR OCCURRED IN A SUBROUTINE, THE ADDRESS OF THE SUBROUTINE CALL IS ALSO GIVEN. REFER TO THE PROGRAM LISTING AT THE STATED ADDRESS TO DETERMINE THE CAUSE OF THE ERROR.

6.1 'FATAL' ERRORS

IN THE EVENT THAT THE DISK DRIVE BECOMES UNAVAILABLE TO THE CONTROLLER, POWERS DOWN, OR CERTAIN CRITICAL STATUS BITS CANNOT BE CLEARED PRIOR TO THE START OF A TEST SEQUENCE - THIS INFORMATION WILL BE COMMUNICATED TO THE OPERATOR. IN ADDITION THE TTY BELL WILL RING AND THE PROGRAM WILL HALT.

IT IS SUGGESTED THAT IF THIS HAPPENS THE OPERATOR LOAD ADDRESS 200 (210) AND RESTART THE PROGRAM AS A FIRST ATTEMPT TO SOLVE THE PROBLEM. IF THE FAILURE CONTINUES TO OCCUR, LOOK IN THE TEST LISTING FOR THE 'HALT' INSTRUCTION AND REPLACE IT PLUS THE TWO WORDS ('TYPE ,CPHALT') ABOVE WITH 'NOP'S. WITH TTY ERROR PRINTOUTS INHIBITED A SCOPE LOOP CAN BE INITIATED FOR THE TEST IN QUESTION.

IT IS ALSO POSSIBLE TO CONTINUE FROM THE HALT POINT, BUT IT IS NOT RECOMMENDED AS ALL FOLLOWING TESTS WILL EXHIBIT THE SAME SYMPTOMS AND GIVE MISLEADING ERROR PRINTOUTS.

7.0 RESTRICTIONS

IF THERE IS A DRIVE CONNECTED THEN THE OPERATOR MUST HAVE THE DRIVE PORT SWITCH LOCKED EITHER ON PORT A OR PORT B BUT NEVER LEAVE IT IN THE PROGRAMMABLE STATE. IF THERE IS NO DRIVE CONNECTED THEN THE CABLE NORMALLY GOING FROM THE DCL TO THE MDLI MUST BE PROPERLY TERMINATED.

SWITCH 12 MUST BE SET WHEN RUNNING ON AN RH70 CONTROLLER AND IT MUST NOT BE SET WHEN RUNNING ON AN RH11 CONTROLLER. BECAUSE OF THE REQUIREMENT FOR IT TO BE SET WHEN USING AN RH70, THE PROGRAM CANNOT BE RUN IN CHAIN MODE WHEN USING THE SOFTWARE SWITCH REGISTER FEATURE WHILE RUNNING ON AN RH70. THIS IS BECAUSE THE ROUTINE WHICH GETS 'SOFTWARE' SWITCH SETTINGS IS NOT OPERABLE WHEN IN CHAIN MODE.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE FIRST PASS OF THE PROGRAM WILL TAKE 30 SECONDS PER DRIVE. SUBSEQUENT PASSES WILL TAKE 1 MINUTE.

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

8.3 OPERATOR SELECTABLE SCOPE LOOPS

HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS. FOR INSTRUCTIONS REGARDING USAGE OF THIS TECHNIQUE, HIT ^C ANY TIME WHILE THE PROGRAM IS RUNNING. ON HITTING AN ERROR IF THE

343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367

LOOP ON ERROR SWITCH IS SET, THE PROGRAM GOES BACK - USUALLY
BACK TO THE BEGINNING OF THE TEST.

WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
THE PROGRAM GOES BACK TO CAN BE CHANGED.
THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
2. LOOP ON ERROR SWITCH MUST BE SET
3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
COMES TO THE END OF THE TEST UNDER CONSIDERATION.

AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
NORMAL OPERATION WILL CONTINUE.

9.0 PROGRAM DESCRIPTION

THE FOLLOWING SECTIONS DESCRIBE EACH TEST AND SUBROUTINES
IN DETAIL AND CAN ALSO BE USED AS AN INDEX TO THE LISTING.
THE LEFT MOST COLUMN IS THE LINE NUMBER WITHIN THE LISTING
WHERE THAT ITEM WILL BE FOUND.

a

1
547
548

549

550
551
552
553

```

;*LAST REVISION 05-NOV-81

.TITLE CZRJGEO RP04/5/6 DSKLS PT1
;*COPYRIGHT (C) 1976,1978,1981
;*DIGITAL EQUIPMENT CORPORATION
;*COLORADO SPGS., CO. 80919
;*
;*PROGRAM BY PETE BLACKSTONE
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
;*
.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH          USE
;*      -----          -
;*      15              HALT ON ERROR
;*      14              LOOP ON TEST
;*      13              INHIBIT ERROR TYPEOUTS
;*      12              RH70 CONTROLLER SELECT
;*      11              INHIBIT ITERATIONS
;*      10              BELL ON ERROR
;*      9               LOOP ON ERROR
;*      8               LOOP ON TEST IN SWR<7:0>
;*      7               STOP FURTHER COMPARES IF SW08 IS LOW
;*      6               ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100  STACK = 1100
104000  ERROR = EMT          ;;BASIC DEFINITION OF ERROR CALL
000004  SCOPE = IOT         ;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS
000011  HT = 11            ;;CODE FOR HORIZONTAL TAB
000012  LF = 12            ;;CODE FOR LINE FEED
000015  CR = 15            ;;CODE FOR CARRIAGE RETURN
000200  CRLF = 200         ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776  PS = 177776       ;;PROCESSOR STATUS WORD
177776  PSW=PS
177774  STKLMT = 177774    ;;STACK LIMIT REGISTER
177772  PIRQ = 177772     ;;PROGRAM INTERRUPT REQUEST REGISTER
177570  DSWR = 177570     ;;HARDWARE SWITCH REGISTER
177570  DDISP = 177570    ;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS
000000  R0 = %0           ;;GENERAL REGISTER
000001  R1 = %1           ;;GENERAL REGISTER
000002  R2 = %2           ;;GENERAL REGISTER
000003  R3 = %3           ;;GENERAL REGISTER
000004  R4 = %4           ;;GENERAL REGISTER
000005  R5 = %5           ;;GENERAL REGISTER
000006  R6 = %6           ;;GENERAL REGISTER
000007  R7 = %7           ;;GENERAL REGISTER
000006  SP = %6           ;;STACK POINTER
000007  PC = %7           ;;PROGRAM COUNTER

```



```
000000      ;*PRIORITY LEVEL DEFINITIONS
000040      PR0      = 0          ;:PRIORITY LEVEL 0
000100      PR1      = 40         ;:PRIORITY LEVEL 1
000140      PR2      = 100        ;:PRIORITY LEVEL 2
000200      PR3      = 140        ;:PRIORITY LEVEL 3
000240      PR4      = 200        ;:PRIORITY LEVEL 4
000300      PR5      = 240        ;:PRIORITY LEVEL 5
000340      PR6      = 300        ;:PRIORITY LEVEL 6
          PR7      = 340        ;:PRIORITY LEVEL 7
```

```
100000      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
040000      SW15     = 100000
020000      SW14     = 40000
010000      SW13     = 20000
004000      SW12     = 10000
002000      SW11     = 4000
001000      SW10     = 2000
000400      SW09     = 1000
000200      SW08     = 400
000100      SW07     = 200
000040      SW06     = 100
000020      SW05     = 40
000010      SW04     = 20
000004      SW03     = 10
000002      SW02     = 4
000001      SW01     = 2
          SW00     = 1
001000      SW9=SW09
000400      SW8=SW08
000200      SW7=SW07
000100      SW6=SW06
000040      SW5=SW05
000020      SW4=SW04
000010      SW3=SW03
000004      SW2=SW02
000002      SW1=SW01
000001      SW0=SW00
```

```
100000      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
040000      BIT15    = 100000
020000      BIT14    = 40000
010000      BIT13    = 20000
004000      BIT12    = 10000
002000      BIT11    = 4000
001000      BIT10    = 2000
000400      BIT09    = 1000
000200      BIT08    = 400
000100      BIT07    = 200
000040      BIT06    = 100
000020      BIT05    = 40
000010      BIT04    = 20
000004      BIT03    = 10
000002      BIT02    = 4
000001      BIT01    = 2
          BIT00     = 1
001000      BIT9=BIT09
```

000400 BIT8=BIT08
 000200 BIT7=BIT07
 000100 BIT6=BIT06
 000040 BIT5=BIT05
 000020 BIT4=BIT04
 000010 BIT3=BIT03
 000004 BIT2=BIT02
 000002 BIT1=BIT01
 000001 BIT0=BIT00

000004 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
 ERRVEC = 4 ;:TIME OUT AND OTHER ERRORS
 000010 RESVEC = 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
 000014 TBITVEC = 14 ;: "T" BIT
 000014 TRTVEC = 14 ;:TRACE TRAP
 000014 BPTVEC = 14 ;:BREAKPOINT TRAP (BPT)
 000020 IOTVEC = 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
 000024 PWRVEC = 24 ;:POWER FAIL
 000030 EMTVEC = 30 ;:EMULATOR TRAP (EMT) **ERROR**
 000034 TRAPVEC = 34 ;: "TRAP" TRAP
 000060 TKVEC = 60 ;:TTY KEYBOARD VECTOR
 000064 TPVEC = 64 ;:TTY PRINTER VECTOR
 000240 PIRQVEC = 240 ;:PROGRAM INTERRUPT REQUEST VECTOR

554
 555
 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

.SBTTL RH11/RH70 REGISTERS

;*WORD COUNT REGISTER (RHWC)
 ;*EACH BIT IS CALLED BY BIT NUMBER

;*BUS ADDRESS REGISTER (RHBA)
 ;*EACH BIT IS CALLED BY BIT NUMBER

;*CONTROL AND STATUS REGISTER 2 (RHCS2)

000001 US1= 1 ;:UNIT SELECT (BIT #0)
 000002 US2= 2 ;:UNIT SELECT (BIT #1)
 000004 US4= 4 ;:UNIT SELECT (BIT #2)
 000010 BAI= 10 ;:BUS ADDRESS INCREMENT INHIBIT (BIT #3)
 000020 PAT= 20 ;:INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
 000040 CLR= 40 ;:CLEAR (BIT #5)
 000100 IR= 100 ;:INPUT READY (BIT #6)
 000200 OR= 200 ;:OUTPUT READY (BIT #7)
 000400 MPE= 400 ;:MASS BUS PARITY ERROR (BIT #8)
 001000 MXF= 1000 ;:MISSED TRANSFER ERROR (BIT #9)
 002000 PGE= 2000 ;:PROGRAM ERROR (BIT #10)
 004000 NEM= 4000 ;:NON EXISTANT MEMORY (BIT #11)
 010000 NED= 10000 ;:NON EXISTANT DRIVE (BIT #12)
 020000 UPE= 20000 ;:UNIBUS PARITY ERROR (BIT #13)
 040000 WCE= 40000 ;:WRITE CHECK ERROR (BIT #14)
 100000 DLT= 100000 ;:DATA LATE (BIT #15)

;*DATA BUFFER REGISTER (RHDB)
 ;*EACH BIT IS CALLED BY BIT NUMBER

```

588
589      ;*RP04 REGISTERS
590
591      ;*CONTROL AND STATUS 1 REGISTER. (#00)
592
593      000001      GO=      1      ;GO (BIT #0)
594      000100      IE=      100     ;INTERRUPT ENABLE (BIT #6)
595      000200      RDY=     200     ;READY (BIT #7)
596      000400      A16=     400     ;HIGH ORDER UNIBUS BITS (BIT #8)
597      001000      A17=    1000     ;HIGH ORDER UNIBUS BITS (BIT #9)
598      002000      PSEL=    2000     ;PORT SELECT (BIT #10)
599      004000      DVA=     4000     ;DEVICE AVAILABLE (BIT #11)
600      020000      MCPE=   20000     ;MASSBUSS PARITY ERROR (BIT #13)
601      040000      TRE=     40000     ;TRANSFER ERROR (BIT #14)
602      100000      SC=     100000    ;SPECIAL CONDITION (BIT #15)
603
604      ;*STATUS REGISTER (RHDS1) (#01)
605
606      000001      DFB=      1      ;DRIVE FORWARD 5'/SEC. (BIT #0)
607      000002      DFF20=  2      ;DRIVE FORWARD 20'/SEC. (BIT #1)
608      000004      DIGB=   4      ;DRIVE TO INNER GAVRD BAND (BIT #2)
609      000010      GRV=    10      ;GO REVERSE (BIT #3)
610      000020      DL64=   20      ;DIFFERENCE LESS THAN 64 (BIT #4)
611      000040      DE1=    40      ;DIFFERENCE EQUALS 1 (BIT #5)
612      000100      VV=     100     ;VOLUME VALID (BIT #6)
613      000200      DRY=    200     ;DRIVE READY (BIT #7)
614      000400      DPR=    400     ;DRIVE PRESENT (BIT #8)
615      001000      PROG=   1000     ;PROGRAMABLE (BIT #9)
616      002000      LST=    2000     ;LAST SECTOR TRANSFERRED (BIT #10)
617      004000      WRL=    4000     ;WRITE LOCK (BIT #11)
618      010000      MOL=   10000     ;MEDIUM ON-LINE (BIT #12)
619      020000      PIP=   20000     ;POSITIONING OPERATION IN PROGRESS (BIT #13)
620      040000      ERR=   40000     ;COMPOSIT ERROR. (BIT #14)
621      100000      ATA=  100000     ;ATTENTION ACTIVE (BIT #15)
622
623      ;*ERROR REGISTER #01 (RHER1) (#02)
624
625      000001      ILF=      1      ;ILLEGAL FUNCTION (BIT #0)
626      000002      ILR=      2      ;ILLEGAL REGISTER (BIT #1)
627      000004      RMR=      4      ;REGISTER MODIFICATION REFUSED (BIT #2)
628      000010      PAR=     10      ;PARITY ERROR (BIT #3)
629      000020      FER=     20      ;FORMAT ERROR (BIT #4)
630      000040      WCF=     40      ;WRITE CLOCK FAIL (BIT #5)
631      000100      ECH=    100     ;ECC HARD ERROR (BIT #6)
632      000200      HCE=    200     ;HEADER COMPARE ERROR (BIT #7)
633      000400      HCRC=   400     ;HEADER CRC ERROR (BIT #8)
634      001000      AOE=   1000     ;ADDRESS OVERFLOW ERROR (BIT #9)
635      002000      IAE=   2000     ;INVALID ADDRESS ERROR (BIT #10)
636      004000      WLE=   4000     ;WRITE LOCK ERROR (BIT #11)
637      010000      DTE=  10000     ;DRIVE TIMING ERROR (BIT #12)
638      020000      OPI=  20000     ;OPERATION INCOMPLETE (BIT #13)
639      040000      UNS=  40000     ;DRIVE UNSAFE (BIT #14)
640      100000      DCK= 100000     ;DATA CHECK ERROR (BIT 15)
641
642      ;*MAINTAINABILITY REGISTER (RHMR) (#03)
643
644      000001      DMD=      1      ;DIAGINOSTIC MODE (BIT #0)
  
```


645	000002	MCLK= 2	:MAINTAINABILITY CLOCK (BIT #1)
646	000004	MINX= 4	:MAINTAINABILITY INDEX (BIT #2)
647	000010	MSTCK= 10	:MAINTAINABILITY SECTOR CLOCK (BIT #3)
648	000020	MRD= 20	:MAINTAINABILITY READ (BIT #4)
649	000040	MWR= 40	:MAINTAINABILITY WRITE (BIT #5)
650	000200	DENVL= 200	:DATA ENVELOPE (BIT #7)
651	000400	ZER= 400	:ZERO DETECT (BIT #8)
652	001000	DTSY= 1000	:MAINTAINABILITY SYNC DETECTED (BIT #9)
653			
654		:*ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)	
655			
656	000001	AT0= 1	:DEVICE 0 (BIT #0)
657	000002	AT1= 2	:DEVICE 1 (BIT #1)
658	000004	AT2= 4	:DEVICE 2 (BIT #2)
659	000010	AT3= 10	:DEVICE 3 (BIT #3)
660	000020	AT4= 20	:DEVICE 4 (BIT #4)
661	000040	AT5= 40	:DEVICE 5 (BIT #5)
662	000100	AT6= 100	:DEVICE 6 (BIT #6)
663	000200	AT7= 200	:DEVICE 7 (BIT #7)
664			
665			
666		:*DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)	
667		:*EACH BIT IS CALLED BY BIT NUMBER	
668			
669			
670		:*DRIVE TYPE REGISTER (RHDT) (#06)	
671		:*EACH BIT IS CALLED BY BIT NUMBER	
672			
673			
674		:*LOOK-AHEAD REGISTER (RHLA) (#07)	
675			
676	000001	EXT1= 1	:EXTENSION 1 (BIT #0)
677	000002	EXT2= 2	:EXTENSION 2 (BIT #1)
678	000004	EXT4= 4	:EXTENSION 3 (BIT #2)
679	000010	EXT10= 10	:EXTENSION 4 (BIT #3)
680	000020	EXT20= 20	:EXTENSION 5 (BIT #4)
681	000040	EXT40= 40	:EXTENSION 6 (BIT #5)
682	000100	SC1= 100	:SECTOR COUNT FIELD 0 (BIT #6)
683	000200	SC2= 200	:SECTOR COUNT FIELD 1 (BIT #7)
684	000400	SC4= 400	:SECTOR COUNT FIELD 2 (BIT #8)
685	001000	SC10= 1000	:SECTOR COUNT FIELD 3 (BIT #9)
686	002000	SC20= 2000	:SECTOR COUNT FIELD 4 (BIT #10)
687	004000	TRK1= 4000	:TRACK FIELD 1 (BIT #11)
688	010000	TRK2= 10000	:TRACK FIELD 2 (BIT #12)
689	020000	TRK4= 20000	:TRACK FIELD 3 (BIT #13)
690	040000	TRK10= 40000	:TRACK FIELD 4 (BIT #14)
691	100000	TRK20= 100000	:TRACK FIELD 5 (BIT #15)
692			
693		:*RP04 ERROR REGISTER #2 (RHER2) (#10)	
694			
695	000001	WCU= 1	:WRITE CURRENT UNSAFE (BIT #0)
696	000002	CSF= 2	:CURRENT SINK FAILURE (BIT #1)
697	000004	WSU= 4	:WRITE SELECT UNSAFE (BIT #2)
698	000010	CSU= 10	:CURRENT SWITCH UNSAFE (BIT #3)
699	000020	MSE= 20	:MOTOR SEQUENCE ERROR (BIT #4)
700	000040	TDF= 40	:TRANSITIONS DETECTOR FAILURE (BIT #5)
701	000100	TUF= 100	:TRANSITIONS UNSAFE (BIT #6)

702	000200	FEN= 200	: FAILSAFE ENABLED (BIT #7)
703	000400	WRU= 400	: WRITE READY UNSAFE (BIT #8)
704	001000	MHS= 1000	: MULTIPLE HEAD SELECT (BIT #9)
705	002000	NHS= 2000	: NO HEAD SELECTION (BIT #10)
706	004000	IXE= 4000	: INDEX ERROR (BIT #11)
707	010000	VU30= 10000	: 30VOLT UNSAFE (BIT #12)
708	020000	PLU= 20000	: PLO UNSAFE (BIT #13)
709	100000	ACU= 100000	: ACUNSAFE (BIT #15)
710			
711		:*RP05/6 ERROR REGISTER #2 (RHER2) (#10)	
712			
713	000001	WCU= 1	: WRITE CURENT UNSAFE
714	000002	CSF= 2	: CURRENT SINK FAILURE
715	000004	WSU= 4	: CURENT SELECT UNSAFE
716	000010	CSU= 10	: CURRENT SWITCH UNSAFE
717	000020	RAW= 20	: READ AND WRITE
718	000040	TDF= 40	: TRANSITIONS DETECTOR FAILURE
719	000100	TUF= 100	: TRANSITIONS UNSAFE
720	000200	ABS= 200	: ABNORMAL STOP
721	000400	WRU= 400	: WRITE READY UNSAFE
722	001000	MHS= 1000	: MULTIPLE HEAD SELECT
723	002000	NHS= 2000	: NO HEAD SELECTION
724	004000	IXE= 4000	: INDEX ERROR
725	020000	PLU= 20000	: PLO UNSAFE
726			
727		:*OFFSET REGISTER (RHOF) (#11)	
728			
729	000001	OF25= 1	: OFFSET 25 MICRO INCHES (BIT #0)
730	000002	OF50= 2	: OFFSET 50 MICRO INCHES (BIT #1)
731	000004	OF100= 4	: OFFSET 100 MICRO INCHES (BIT #2)
732	000010	OF200= 10	: OFFSET 200 MICRO INCHES (BIT #3)
733	000020	OF400= 20	: OFFSET 400 MICRO INCHES (BIT #4)
734	000040	OF800= 40	: OFFSET 800 MICRO INCHES (BIT #5)
735			
736	000200	OFREV= 200	: OFFSET NEGATIVE (REVERSE) (BIT #7)
737	002000	HCI= 2000	: HEADER COMPARE INHIBIT (BIT #10)
738	004000	ECI= 4000	: ERROR CORRECTION CODE INHIBIT (BIT #11)
739	010000	FMT22= 10000	: FORMAT BIT (BIT #12)
740			
741		:*DESIRED CYLINDER ADDRESS (RHCA) (#12)	
742		:*EACH BIT IS CALLED BY BIT NUMBER.	
743			
744		:*CURRENT CYLINDER ADDRESS (RHCC) (#13)	
745		:*EACH BIT IS CALLED BY BIT NUMBER	
746			
747		:*SERIAL NUMBER REGISTER (RHSN) (#14)	
748		:*EACH IS CALLED BY BIT NUMBER	
749			
750		:*ERROR REGISTER #03 (RHER3) (#15)	
751			
752			
753			
754			
755			
756	000001	PSU= 1	: PACK SPEED UNSAFE (BIT #0)
757	000002	VUF= 2	: VELOCITY UNSAFE (BIT #1)
758	000010	UWR= 10	: ANY UNSAFE EXCEPT READ/WRITE (BIT #3)

```
759      000020      PRE=   20      ;DISK PACK ROTATION ERROR (BIT #4)
760      000040      ACL=   40      ;AC LOW (BIT #5)
761      000100      OPDCL= 100      ;DC LOW (BIT #6)
762      040000      SKI=  40000     ;SEEK INCOMPLETE (BIT #14)
763      100000      OCYL= 100000    ;OFF CYLINDER (BIT #15)
764
765
766      ;*ECC POSITION REGISTER (RHEC1) (#16)
767      ;*EACH BIT IS CALLED BY BIT NUMBER
768
769
770      ;*ECC PATTERN REGISTER (RHEC2) (#17)
771      ;*EACH BIT IS CALLED BY BIT NUMBER
772
773      .SBTTL MEMORY MANAGEMENT DEFINITIONS
      ;*KT11 VECTOR ADDRESS
      000250      MMVEC   = 250
      ;*KT11 STATUS REGISTER ADDRESSES
      177572      SR0     = 177572
      177574      SR1     = 177574
      177576      SR2     = 177576
      172516      SR3     = 172516
      ;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
      172300      KIPDR0  = 172300
      172302      KIPDR1  = 172302
      172304      KIPDR2  = 172304
      172306      KIPDR3  = 172306
      172310      KIPDR4  = 172310
      172312      KIPDR5  = 172312
      172314      KIPDR6  = 172314
      172316      KIPDR7  = 172316
      ;*KERNEL 'I' PAGE ADDRESS REGISTERS
      172340      KIPAR0  = 172340
      172342      KIPAR1  = 172342
      172344      KIPAR2  = 172344
      172346      KIPAR3  = 172346
      172350      KIPAR4  = 172350
      172352      KIPAR5  = 172352
      172354      KIPAR6  = 172354
      172356      KIPAR7  = 172356
```

774

C
E

1
2
3
4
5
6

```
.SBTTL TRAP CATCHER
      .=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
      .=174
DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
```

```
.SBTTL STARTING ADDRESS(ES)
      JMP @#BEGIN      ;;JUMP TO STARTING ADDRESS OF PROGRAM
      JMP @#BASECH     ;;MODIFY ADDRESSES
      JMP @#BEGIN2     ;;SELECT DRIVE START
```

```
.SBTTL ACT11 HOOKS
;*****
;HOOKS REQUIRED BY ACT11
      $SVPC=.          ;SAVE PC
      .=46
      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
      .=52
      .WORD 20000     ;;2)SET LOC.52 TO 20000
      .=$SVPC        ;;RESTORE PC
```

```
000000
000174 000174
000174 000000
000176 000000
000200 000137 004310
000204 000137 045104
000210 000137 004316
000046 000214
000046 000046
000046 040252
000052 000052
000052 020000
000052 000214
```

COMMON TAGS

0

.SBTTL COMMON TAGS

::*****
 ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 ::*USED IN THE PROGRAM.

001100	001100			SCMTAG: .WORD	0	:: START OF COMMON TAGS
001100	000000			\$PASS: .WORD	0	:: CONTAINS PASS COUNT
001102	000			\$TSTNM: .BYTE	0	:: CONTAINS THE TEST NUMBER
001103	000			\$ERFLG: .BYTE	0	:: CONTAINS ERROR FLAG
001104	000000			\$ICNT: .WORD	0	:: CONTAINS SUBTEST ITERATION COUNT
001106	000000			\$LPADR: .WORD	0	:: CONTAINS SCOPE LOOP ADDRESS
001110	000000			\$LPERR: .WORD	0	:: CONTAINS SCOPE RETURN FOR ERRORS
001112	000000			\$ERTTL: .WORD	0	:: CONTAINS TOTAL ERRORS DETECTED
001114	000			\$ITEMB: .BYTE	0	:: CONTAINS ITEM CONTROL BYTE
001115	001			\$ERMAX: .BYTE	1	:: CONTAINS MAX. ERRORS PER TEST
001116	000000			\$ERRPC: .WORD	0	:: CONTAINS PC OF LAST ERROR INSTRUCTION
001120	000000			\$GDADR: .WORD	0	:: CONTAINS ADDRESS OF 'GOOD' DATA
001122	000000			\$BDADR: .WORD	0	:: CONTAINS ADDRESS OF 'BAD' DATA
001124	000000			\$GDDAT: .WORD	0	:: CONTAINS 'GOOD' DATA
001126	000000			\$BDDAT: .WORD	0	:: CONTAINS 'BAD' DATA
001130	000000				0	:: RESERVED--NOT TO BE USED
001132	000000				0	
001134	000			\$AUTOB: .BYTE	0	:: AUTOMATIC MODE INDICATOR
001135	000			\$INTAG: .BYTE	0	:: INTERRUPT MODE INDICATOR
001136	000000				0	
001140	177570			\$SWR: .WORD	DSWR	:: ADDRESS OF SWITCH REGISTER
001142	177570			\$DISPLAY: .WORD	DDISP	:: ADDRESS OF DISPLAY REGISTER
001144	177560			\$TKS: 177560		:: TTY KBD STATUS
001146	177562			\$TKB: 177562		:: TTY KBD BUFFER
001150	177564			\$TPS: 177564		:: TTY PRINTER STATUS REG. ADDRESS
001152	177566			\$TPB: 177566		:: TTY PRINTER BUFFER REG. ADDRESS
001154	000			\$NULL: .BYTE	0	:: CONTAINS NULL CHARACTER FOR FILLS
001155	002			\$FILLS: .BYTE	2	:: CONTAINS # OF FILLER CHARACTERS REQUIRED
001156	012			\$FILLC: .BYTE	12	:: INSERT FILL CHARS. AFTER A 'LINE FEED'
001157	000			\$TPFLG: .BYTE	0	:: 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
001160	000000			\$REGAD: .WORD	0	:: CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
001162	000000			\$REG0: .WORD	0	:: CONTAINS ((\$REGAD)+0)
001164	000000			\$REG1: .WORD	0	:: CONTAINS ((\$REGAD)+2)
001166	000000			\$REG2: .WORD	0	:: CONTAINS ((\$REGAD)+4)
001170	000000			\$REG3: .WORD	0	:: CONTAINS ((\$REGAD)+6)
001172	000000			\$REG4: .WORD	0	:: CONTAINS ((\$REGAD)+10)
001174	000000			\$REG5: .WORD	0	:: CONTAINS ((\$REGAD)+12)
001176	000000			\$TMP0: .WORD	0	:: USER DEFINED
001200	000000			\$TMP1: .WORD	0	:: USER DEFINED
001202	000000			\$TMP2: .WORD	0	:: USER DEFINED
001204	000000			\$TMP3: .WORD	0	:: USER DEFINED
001206	000000			\$TMP4: .WORD	0	:: USER DEFINED
001210	000000			\$TMP5: .WORD	0	:: USER DEFINED
001212	000000			\$TIMES: 0		:: MAX. NUMBER OF ITERATIONS
001214	000000			\$ESCAPE: 0		:: ESCAPE ON ERROR ADDRESS
001216	207	377	377	\$BELL: .ASCIZ	<207><377><377>	:: CODE FOR BELL
001222	077			\$QUES: .ASCII	/?/	:: QUESTION MARK
001223	015			\$CRLF: .ASCII	<15>	:: CARRIAGE RETURN
001224	012	000		\$LF: .ASCIZ	<12>	:: LINE FEED

;;*****

.SBTTL USER DEFINED TAGS

001226	000254	RPVEC:	254	:	RP04 VECTOR ADDRESS
001230	176722	RHDB:	176722	:	DATA BUFFER SEE NOTE ABOVE
001232	176702	RHWC:	176702	:	WORD COUNT SEE NOTE ABOVE
001234	176704	RHBA:	176704	:	BUS ADDRESS SEE NOTE ABOVE
001236	176710	RHCS2:	176710	:	CONTROL AND STATUS 2 SEE NOTE ABOVE
001240	176700	RHCS1:	176700	:	CONTROL AND STATUS 1 SEE NOTE ABOVE
001242	176714	RHER1:	176714	:	ERROR #1 SEE NOTE ABOVE
001244	176706	RHDST:	176706	:	DESIRED SECTOR/TRACK ADDRESS SEE NOTE ABOVE
001246	176740	RHER2:	176740	:	ERROR #2 SEE NOTE ABOVE
001250	176732	RHOF:	176732	:	OFFSET SEE NOTE ABOVE
001252	176734	RHCA:	176734	:	DESIRED CYLINDER ADDRESS SEE NOTE ABOVE
001254	176742	RHER3:	176742	:	ERROR #3 SEE NOTE ABOVE
001256	176716	RHAS:	176716	:	ATTENTION SUMMARY SEE NOTE ABOVE
001260	176724	RHMR:	176724	:	MAINTAINABILITY SEE NOTE ABOVE
001262	176712	RHDS1:	176712	:	DRIVE STATUS SEE NOTE ABOVE
001264	176726	RHDT:	176726	:	DRIVE TYPE SEE NOTE ABOVE
001266	176730	RHSN:	176730	:	SERIAL NUMBER SEE NOTE ABOVE
001270	176744	RHEC1:	176744	:	ECC POSITION SEE NOTE ABOVE
001272	176746	RHEC2:	176746	:	ECC PATTERN SEE NOTE ABOVE
001274	176720	RHLA:	176720	:	LOOK-AHEAD SEE NOTE ABOVE
001276	176736	RHCC:	176736	:	CURRENT CYLINDER ADDRESS SEE NOTE ABOVE

:ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER LOGIC

001300	176750	RHBAE:	176750	:	BUS ADDRESS EXTENSION REGISTER
001302	176752	RHCS3:	176752	:	CONTROL AND STATUS REGISTER #3

:THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
:ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
:ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
:FOR THE TIME JUST AFTER THE 'ERROR' ERROR COMMAND

001304	000000	DB:	0	:	DATA BUFFER
001306	000000	WC:	0	:	WORD COUNT
001310	000000	BA:	0	:	BUS ADDRESS
001312	000000	CS2:	0	:	CONTROL AND STATUS 2
001314	000000	CS1:	0	:	CONTROL AND STATUS 1
001316	000000	ER1:	0	:	ERROR #1
001320	000000	DST:	0	:	DESIRED SECTOR/TRACK ADDRESS
001322	000000	ER2:	0	:	ERROR #2
001324	000000	OF:	0	:	OFFSET
001326	000000	CA:	0	:	DESIRED CYLINDER ADDRESS
001330	000000	ER3:	0	:	ERROR #3
001332	000000	AS:	0	:	ATTENTION SUMMARY
001334	000000	MR:	0	:	MAINTAINABILITY
001336	000000	DS1:	0	:	DRIVE STATUS
001340	000000	DT:	0	:	DRIVE TYPE
001342	000000	SN:	0	:	SERIAL NUMBER
001344	000000	EC1:	0	:	ECC POSITION
001346	000000	EC2:	0	:	ECC PATTERN
001350	000000	LA:	0	:	LOOK-AHEAD
001352	000000	CC:	0	:	CURRENT CYLINDER ADDRESS

:FLAGS & INTERNAL PROGRAM CONTROL WORDS

001354		UNITS: .BLKW	8.	:TABLE OF DRIVES PRESENT TO TEST
001374	000000	UNIT: .WORD	0	:UNIT UNDER TEST
001376	000000	NOUNIT: .WORD	0	:NUMBER OF UNITS PRESENT
001400	000000	NUNIT: .WORD	0	:USED TO KEEP TRACK OF UNIT UNDER TEST
001402	000000	SELECT: .WORD	0	:USED TO DETERMINE IF THERE IS MORE
001404	000000	UNITSL: .WORD	0	:THAN ONE UNIT
001406	000000	ERFLGS: 0		:ALL ONES INDICATE UNIT TO BE SELECTED
001410	000000	SAVDT: 0		:UNIT NO. SELECTED
				:ERROR FLAG
				:SAVE DRIVE TYPE REGISTER
				:FOR COMPARISON IN DRIVE CLEAR TEST
				:AND RH INIT TEST
001412	000000	SAVSN: 0		:SAVE SERIAL NUMBER REGISTER
				:FOR COMPARISON IN DRIVE CLEAR TEST
				:AND RH INIT TEST
001414	000000	PCJSR: 0		:SAVE PC OF JSR WHICH GAVE THE ERROR
001416	000000	ATTENT: 0		:ATTENTION BIT FOR PRESENT UNIT
001420	000000	TOTALAT: 0		:TATAL ATTENTION BITS
001422	000000	TMPILL: 0		:TEMPORARY ILLEGAL FUNCTION
001424	000000	TSECC: 0		:FLAG TO SAY IF ECC TEST OR NOT
				:WHEN =177777 IT IS AN ECC TEST
				:WHEN =0IT IS NOT AN ECC TEST
001426	000000	TESDTE: 0		:FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
				:WHEN = 177777 IT IS A DTE TEST
				:WHEN = 0 IT IS NOT A DTE TEST
001430	000000	TAGDTE: 0		:TEMPORARY TAG USED IN DRIVE TIMING
				:ERROR TEST
001432	000000	TSTNM: 0		:TEST NUMBER
001434	000000	RP06: 0		:IF 0 PROGRAM WILL TREAT DRIVE AS RP04
001436	000000	RH70: 0		:IF 1 PROGRAM IS RUNNING ON RH70
				:IF 0 PROGRAM IS ON AN RH11
001440	000000	SILOSZ: .WORD	0	:RH SILO SIZE

:FUNCTION EQUATES

:TABLE OF FUNCTIONS FOR RHCS1, THEN 'GO' BIT HAS TO BE SET

001442		FUTABL:		
001442	000000	NOPERA: 0		:NO OPERATION
001444	000002	UNLOAD: 2		:UNLOAD (STAND BY)
001446	000006	RECALI: 6		:RECALIBRATE
001450	000010	DCLEAR: 10		:DRIVE CLEAR
001452	000012	RELEAS: 12		:RELEASE (DUAL-PORT OPERATION)
001454	000030	SERCH: 30		:SEARCH COMMAND
001456	000050	WRCHK: 50		:WRITE CHECK DATA
001460	000052	WRCHDT: 52		:WRITE CHECK HEADER AND DATA
001462	000060	WRIDAT: 60		:WRITE DATA
001464	000062	WRIFOR: 62		:WRITE HEADER AND DATA (FORMAT)
001466	000070	READAT: 70		:READ DATA
001470	000072	REFOR: 72		:READ HEADER AND DATA
001472	000004	SEECOM: 4		:SEEK COMMAND
001474	000014	OFSETC: 14		:OFFSET COMMAND
001476	000016	RETCL: 16		:RETURNTO CENTERLINE
001500	000022	PKACK: 22		:PACK ACKNOWLEDGE
001502	000020	READIN: 20		:READ IN
001504	000000	ILLEGL: .WORD		:COMPUTED ILLEGAL FUNCTION

;DATA BUFFERS FOR READ WRITE

001510
002554

WRFROM: .BLKW 274.
REINTO: .BLKW 274.

;WRITE FROM THIS BUFFER
;READ INTO THIS BUFFER

;TABLE FOR ATTENTION BITS ATTENTION TABLE

003620

001

002

004

ATABLE: .BYTE 1,2,4,10,20,40,100,200

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 ;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 ;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ::POINTS TO THE ERROR MESSAGE
 ;* DH ::POINTS TO THE DATA HEADER
 ;* DT ::POINTS TO THE DATA
 ;* DF ::POINTS TO THE DATA FORMAT

	003630		\$ERRTB:	
1			:ERROR 1	
2				
3				
4	003630	057072	EM1	:WRONG DATA IN READING OR WRITING HARDWARE REGISTER
5	003632	063205	DH1	:PC
6				:REG. ADDR.
7				:GOOD DATA
8				:RECEIVED DATA
9	003634	067540	DT1	:\$ERRPC,REGADR,\$GDDAT,\$BDDAT
10	003636	070256	DF1	:0.0,0.0,0
11				
12			:ERROR 2	
13				
14	003640	057155	EM2	:ERROR ON DATA COMMAND
15				
16	003642	066323	DH33	:PC
17				:PC OF JSR
18				:TEST NO
19				:WORD NO.
20				:GOOD DATA
21				:CONTENTS OF RHCS1
22				:CONTENTS OF RHDS1
23				:CONTENTS OF RHER1
24	003644	070120	DT33	:\$ERRPC,PCJSR,\$STSTM,ERWORD,\$GDDAT,CS1,DS1,ER1
25	003646	070426	DF33	:0.0,0.1,0.0,0.0
26				
27			:ERROR 3	
28				
29	003650	057155	EM2	:ERROR ON DATA COMMAND
30				
31	003652	066101	DH32	:PC
32				:PC OF JSR
33				:TEST NO
34				:WORD NO.
35				:GOOD DATA
36				:BAD DATA
37				:CONTENTS OF RHCS1
38				:CONTENTS OF RHDS1
39				:CONTENTS OF RHER1
40				
41	003654	070074	DT32	:\$ERRPC,PCJSR,\$STSTM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
42	003656	070415	DF32	:0.0,0.1,0.0,0.0,0.0

43				
44			:ERROR 4	
45				
46	003660	057155	EM2	:ERROR ON DATA COMMAND
47				
48	003662	065677	DH31	:PC
49				:TEST NO
50				:WORD NO.
51				:GOOD DATA
52				:BAD DATA
53				:CONTENTS OF RHCS1
54				:CONTENTS OF RHDS1
55				:CONTENTS OF RHER1
56				
57	003664	070052	DT31	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
58	003666	070405	DF31	:0,0,1,0,0,0,0,0,
59				
60			:ERROR 5	
61				
62	003670	000000	0	:
63	003672	000000	0	:
64	003674	070052	DT31	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
65	003676	070405	DF31	:0,0,1,0,0,0,0,0,
66				
67			:ERROR 6	
68				
69	003700	057204	EM6	:ERROR ON WRITE HEADER AND DATA
70				
71	003702	066101	DH32	:PC
72				:PC OF JSR
73				:TEST NO
74				:WORD NO.
75				:GOOD DATA
76				:BAD DATA
77				:CONTENTS OF RHCS1
78				:CONTENTS OF RHDS1
79				:CONTENTS OF RHER1
80				
81	003704	070074	DT32	:\$ERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
82	003706	070415	DF32	:0,0,0,1,0,0,0,0,0,
83				
84			:ERROR 7	
85				
86	003710	057204	EM6	:ERROR ON WRITE HEADER AND DATA
87	003712	063327	DH2	:PC
88				:TEST NO
89				:WORD NO.
90				:GOOD DATA
91				:BAD DATA
92	003714	067566	DT3	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT
93	003716	070267	DF3	:0,0,1,0,0,
94				
95			:ERROR 10	
96				
97	003720	000000	0	:
98	003722	000000	0	:
99	003724	067566	DT3	:\$ERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT

100	003726	070267	DF3	:0,0,1,0,0,
101				
102			:ERROR 11	
103				
104	003730	057243	EM11	:CONTROLLER OR DRIVE STATUS
105	003732	063451	DH11	:PC
106				:TEST NO
107				:FAILING REG. ADDR
108				:CONTENTS OF RHCS1
109				:CONTENTS OF RHCS2
110				:CONTENTS OF RHDS1
111				:CONTENTS OF RHER1
112	003734	067602	DT11	:\$ERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1
113	003736	070274	DF11	:0,0,0,0,0,0
114				
115			:ERROR 12	
116				
117	003740	057243	EM11	:WRONG DATA FROM SILO
118				
119	003742	063205	DH1	:PC
120				:REG.ADDR
121				:GOOD DATA
122				:RECEIVED DATA
123	003744	067540	DT1	:\$ERRPC,REGADR,\$GDDAT,\$BDDAT
124	003746	070256	DF1	:0,0,0,0
125				
126			:ERROR 13	
127				
128	003750	000000	0	
129	003752	000000	0	
130	003754	067540	DT1	:\$ERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT
131	003756	070256	DF1	:0,0,0,0,0
132				
133			:ERROR 14	
134				
135	003760	057276	EM14	:REGISTER FAILED
136	003762	063630	DH14	:PC
137				:FAILING REG. ADDR
138				:CONTENTS OF FAILING REG.
139				:CONTENTS OF RHCS1
140				:CONTENTS OF RHCS2
141				:CONTENTS OF RHDS1
142				:CONTENTS OF RHER1
143	003764	067622	DT14	:\$ERRPC,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1
144	003766	070303	DF14	:0,0,0,0,0,0,0
145				
146			:ERROR 15	
147				
148	003770	057316	EM15	:SPECIFIED REG. NON EXISTANT SO ABORT
149				:PROGRAM
150	003772	064027	DH15	:PC
151				:ADDR. OF REG
152	003774	067644	DT15	:\$ERRPC,TEMP1
153	003776	070313	DF15	:0,0
154				
155			:ERROR 16	
156				

157	004000	057366	EM16	:WAIT LOOP FAILED
158	004002	064060	DH16	:PC
159				:WAT PC
160				:BIT WANTED
161				:REG. ADR.
162				:REG. CONT.
163	004004	067654	DT16	:\$ERRPC,\$TMP3,\$TMP1,\$TMP0,\$BDDAT
164	004006	070316	DF16	:0,0,0,0
165				
166				:ERROR 17
167				
168	004010	057407	EM17	:WRITE CHECK FAILING
169	004012	064222	DH17	:PC
170				:TEST NO
171				:CONTENTS OF RHBA
172				:CONTENTS OF RHDB
173				:CONTENTS OF RHWC
174				:CONTENTS OF RHCS1
175				:CONTENTS OF RHCS2
176	004014	067672	DT17	:\$ERRPC,\$TSTNM,\$BA,DB,WC,CS1,CS2
177	004016	070323	DF17	:0,0,0,0,0,0,0
178				
179				:ERROR 20
180				
181	004020	057433	EM20	:REGISTER FAILING
182	004022	064404	DH20	:PC
183				:TST NO
184				:CONTENTS OF RHER1
185				:CONTENTS OF RHER2
186				:CONTENTS OF RHER3
187				:CONTENTS OF RHAS
188				:CONTENTS OF RHDS1
189	004024	067712	DT20	:\$ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1
190	004026	070332	DF20	:0,0,0,0,0,0,0
191				
192				:ERROR 21
193				
194	004030	057454	EM21	:INTERRUPT FAILING
195	004032	064566	DH21	:PC
196				:TEST NO
197				:CONTENTS OF RHCS1
198				:CONTENTS OF RHAS
199				:CONTENTS OF RHDS1
200	004034	067732	DT21	:\$ERRPC,TSTNM,CS1,AS,DS1
201	004036	070341	DF21	:0,0,0,0,0
202				
203				:ERROR 22
204				
205	004040	057476	EM22	:ERROR IN DRIVE PRESENT -
206				:LOOKING AT RHAS AND RHCS2-NED(BIT#12)
207				:DRIVES PRESENT DO NOT AGREE
208				:NOTE: ON DUAL PORT SYSTEM
209				:DRIVE CN OTHER PORT WILL NOT GIVE NED
210				:HENCE THERE WILL BE A MISSMATCH
211	004042	064705	DH22	:PC
212				:TEST NO
213				:RHAS UNIT (RHER1 BITS SET)

214				:RHCS2 UNIT ('NED' BIT TEST)
215	004044	067746	DT22	:\$ERRPC,TSTNM
216	004046	070346	DF22	:0,0
217				
218			;ERROR 23	
219				
220	004050	000000	0	:NO LONGER USED DUE TO SPECIAL 'NED'
221	004052	000000	0	:TEST TABLE TYPE OUT ROUTINE
222	004054	000000	0	
223	004056	000000	0	
224				
225			;ERROR 24	
226				
227	004060	060147	EM24	:LOOK AHEAD REGISTER AT THE
228				:BEGINNING OF A SECTOR IS IN
229				:ERROR
230	004062	065007	DH24	:PC
231				:RHDST
232				:BAD RHLA
233				:GOOD RHLA
234				:SECTOR NO
235				:SECTOR CLOCK
236	004064	067754	DT24	:\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
237	004066	070352	DF24	:0,0,0,0,0
238				
239			;ERROR 25	
240				
241	004070	060242	EM25	:LOOK AHEAD REGISTER IS
242				:IN ERROR
243				
244	004072	065007	DH24	:PC
245				:RHDST
246				:BAD RHLA
247				:GOOD RHLA
248				:SECTOR NO
249				:SECTOR CLOCK
250	004074	067754	DT24	:\$ERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
251	004076	070352	DF24	:0,0,0,0,0
252				
253			;ERROR 26	
254				
255	004100	057243	EM11	:CONTROLLER OR DRIVE STATUS
256				
257	004102	065171	DH26	:PC
258				:PC OF JSR
259				:FAILING REGISTER ADDRESS
260				:CONTENTS OF RHCS1
261				:CONTENTS OF RHCS2
262				:CONTENTS OF RHDS1
263				:CONTENTS OF RHER1
264				
265	004104	067774	DT26	:\$ERRPC,PCJSR,\$BDADR,CS1,CS2,DS1,ER1
266	004106	070361	DF26	:0,0,0,0,0,0,
267				
268			;ERROR 27	
269				
270	004110	057072	EM1	:ERROR IN READING OR WRITING HARDWARE REGISTER

271				
272	004112	065373	DH27	:PC
273				:PC OF JSR
274				:TEST NUMBER
275				:FAILING REGISTER
276				:GOOD DATA
277				:RECEIVED DATA
278				
279	004114	070016	DT27	:\$ERRPC,PCJSR,TSTNM,REGADR,\$GDDAT,\$BDDAT
280	004116	070371	DF27	:0,0,0,0,0,0
281				
282				:ERROR 30
283				
284	004120	060302	EM30	:CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
285	004122	065535	DH30	:PC
286				:PC OF JSR
287				:REGISTER ADDRESS
288				:GOOD DATA
289				:BAD DATA
290				
291	004124	070034	DT30	:\$ERRPC,PCJSR,REGADR,\$GDDAT,\$BDDAT
292	004126	070377	DF30	:0,0,0,0,0
293				
294				:ERROR 31
295				
296	004130	060422	EM31	:ECC GENERATED IS INCORRECT
297				:EVERY WORD IN THIS SECTOR IS GIVEN IN 'DATA USED'
298				
299	004132	066525	DH34	:PC
300				:TEST NUMBER
301				:GOOD ECC1
302				:GOOD ECC2
303				:WRITTEN ECC1
304				:WRITTEN ECC2
305				:DATA USED
306				
307	004134	070142	DT34	:\$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK
308				
309	004136	070436	DF34	:0,0,0,0,0,0,0
310				
311				:ERROR 32
312				
313	004140	060544	EM32	:ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
314				:ECC REGISTER OR RHER1 IS IN ERROR
315				:ONLY LOWER 11 BITS OF PATTERN REGISTER
316				:CAN BE READ
317				:THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
318				
319	004142	066707	DH35	:PC
320				:TEST NUMBER
321				:GOOD ECC1
322				:GOOD ECC2
323				:PATTERN REGISTER
324				:RHER1
325				
326	004144	070162	DT35	:\$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1
327				

328	004146	070445	DF35	:0,0,0,0,0,0
329				
330			:ERROR 33	
331				
332	004150	061030	EM33	:HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
333	004152	067111	DH36	:PC
334				:PC OF JSR
335				:TEST NUMBER
336				:RHMR
337				:POSITION REG.
338				:PATTERN REGISTER
339				
340	004154	070204	DT36	:\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
341				
342	004156	070455	DF36	:0,0,0,0,0,0
343				
344			:ERROR 34	
345				
346	004160	061102	EM34	:ZERO DETECT BIT NOT HIGH WHEN THE
347				:32 BIT ECC REGISTER HAS ITS 21 BITS
348				:OF ZEROS
349				:ERROR PRINTOUT WILL CONTINUE TILL
350				:ZERO DETECT BIT IS HIGH
351	004162	067111	DH36	:PC
352				:PC OF JSR
353				:TEST NUMBER
354				:RHMR
355				:POSITION REG.
356				:PATTERN REGISTER
357				
358	004164	070204	DT36	:\$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
359				
360	004166	070455	DF36	:0,0,0,0,0,0
361				
362			:ERROR 35	
363				
364	004170	061175	EM35	:POSITION REGISTER OR 11 BITS OF
365				:PATTERN REGISTER INCORRECT
366				:LOWER 11 BITS OF PATTERN REGISTER
367				:SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
368				:DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL
369				
370	004172	067247	DH37	:PC
371				:TEST NUMBER
372				:ECC POSITION
373				:GOOD POSITION
374				:GOOD ECC1
375				:GOOD ECC2
376				:ECC PATTERN
377				:DATA ENVELOPE
378				:N-CODE ZEROS
379				
380	004174	070222	DT37	:\$ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE
381				
382	004176	070463	DF37	:0,0,0,0,0,0,0,0,0
383				
384			:ERROR 36	

385				
386	004200	061471	EM36	:ON A READ COMMAND WITH NON CORRECTABLE
387				:ERROR INSERTED DCK AND ECH SHOULD BE SET
388	004202	066707	DH35	:PC
389				:TEST NUMBER
390				:GOOD ECC1
391				:GOOD ECC2
392				:PATTERN REGISTER
393				:POSITION REGISTER
394				:RHER1
395				
396	004204	070162	DT35	:\$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,ER1
397				
398	004206	070445	DF35	:0,0,0,0,0,0,0
399				
400				:ERROR 37
401				
402	004210	061602	EM37	:PGE ERROR
403	004212	063451	DH11	:PC
404				:TEST NO
405				:FAILING REG. ADDR
406				:CONTENTS OF RHCS1
407				:CONTENTS OF RHCS2
408				:CONTENTS OF RHDS1
409				:CONTENTS OF RHER1
410	004214	067602	DT11	:\$ERRPC,\$TSTNM,\$BDADR,CS1,CS2,DS1,ER1
411	004216	070274	DF11	:0,0,0,0,0,0
412				
413				:ERROR 40
414				
415	004220	061736	EM40	:RHC DID NOT = 0 AFTER A READ OR
416				:WRITE HEADER AND DATA
417	004222	067464	DH40	:PC
418				:TEST NO
419				:CONTENTS OF RHC
420	004224	070246	DT40	:\$ERRPC,TSTNM,\$BDDAT
421	004226	070474	DF40	:0,0,0

```

1      ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
2
3      004230 011600      BADTMO: MOV      (SP),R0      ;SAVE PC WHERE THE TIME OUT OCCURED
4      004232 005740      TST      -(R0)      ;ADJUST PC -2
5      004234 022626      CMP      (SP)+,(SP)+      ;RESTORE STACK POINTER
6      004236 104401 004244      TYPE      ,65$      ;:TYPE ASCIZ STRING
      004242 000417      BR      64$      ;:GET OVER THE ASCIZ
      ;:65$: .ASCIZ <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
      64$:
7      004302 010046      MOV      R0,-(SP)      ;SETUP FOR TYPING OUT PC
8      004304 104402      TYPOC
9      004306 000240      NOP      ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
      ;TO STOP ON UNEXPECTED TIMEOUT.
10
11
12     .SBTTL  START OF PROGRAM
13
14     004310 005037 001402      BEGIN: CLR      SELECT      ;DO NOT SELECT UNIT, NORMAL RUN
15     004314 000403      BR      START
16
17     004316 012737 177777 001402      BEGIN2: MOV     #-1,SELECT      ;SELECT UNIT
18
19     004324 005227 000000      START: INC     #0      ;TTY LOOP, WAIT FOR INCREMENT
20     004330 001375      BNE     .-4      ;OF WORD
21     004332 000005      RESET      ;RESET THE WORLD
22
23     .SBTTL  INITIALIZE THE COMMON TAGS
      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
      MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
      CLR     (R6)+      ;;CLEAR MEMORY LOCATION
      CMP     #SWR,R6      ;;DONE?
      BNE     .-6      ;;LOOP BACK IF NO
      MOV     #STACK,SP      ;;SETUP THE STACK POINTER
      ;;INITIALIZE A FEW VECTORS
      MOV     #$SCOPE,@#IOTVEC      ;;IOT VECTOR FOR SCOPE ROUTINE
      MOV     #340,@#IOTVEC+2      ;;LEVEL 7
      MOV     #$ERROR,@#EMTVEC      ;;EMT VECTOR FOR ERROR ROUTINE
      MOV     #340,@#EMTVEC+2      ;;LEVEL 7
      MOV     #$TRAP,@#TRAPVEC      ;;TRAP VECTOR FOR TRAP CALLS
      MOV     #340,@#TRAPVEC+2      ;;LEVEL 7
      MOV     #$SPWRDN,@#PWRVEC      ;;POWER FAILURE VECTOR
      MOV     #340,@#PWRVEC+2      ;;LEVEL 7
      CLR     $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
      CLR     $ESCAPE      ;;CLEAR THE ESCAPE ON ERROR ADDRESS
      MOVB   #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
      MOV     #,$SLPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
      MOV     #,$SLPERR      ;;SETUP THE ERROR LOOP ADDRESS
      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
      MOV     @#ERRVEC,-(SP)      ;;SAVE ERROR VECTOR
      MOV     #64$,@#ERRVEC      ;;SET UP ERROR VECTOR
      MOV     #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
      MOV     #DDISP,DISPLAY      ;;AND A HARDWARE DISPLAY REGISTER
      CMP     #-1,@SWR      ;;TRY TO REFERENCE HARDWARE SWR
      BNE     66$      ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
      ;AND THE HARDWARE SWR IS NOT = -1
      BR     65$      ;;BRANCH IF NO TIMEOUT
      64$: MOV     #65$,(SP)      ;;SET UP FOR TRAP RETURN
    
```

```

004532 000002          RTI
004534 012737 000176 001140 65$: MOV #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004542 012737 000174 001142      MOV #DISPREG,DISPLAY
004550 012637 000004          66$: MOV (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

24      ;SETUP "TIMEOUT" TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
25 004554 012737 004230 000004      MOV #BADTMO,ERRVEC  ;;SETUP FOR UNEXPECTED TIMEOUT
26 004562 012737 000300 000006      MOV #PR6,ERRVEC+2  ;;LEVEL 6
27
31      .SBTTL TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
004570 005227 177777          INC #-1          ;;FIRST TIME?
004574 001030          BNE 67$          ;;BRANCH IF NO
004576 104401 004604          TYPE ,68$         ;;TYPE ASCIZ STRING
004602 000425          BR 67$          ;;GET OVER THE ASCIZ
      ;;68$: .ASCIZ <CRLF>@CZRJGEO - RP04/5/6 DISKLESS TEST, PT 1@<CRLF>
      67$:
      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
004656 005737 000042          TST @#42         ;;ARE WE RUNNING UNDER XXDP/ACT?
004662 001006          BNE 69$          ;;BRANCH IF YES
004664 023727 001140 000176      CMP SWR,#SWREG   ;;SOFTWARE SWITCH REG SELECTED?
004672 001005          BNE 70$          ;;BRANCH IF NO
004674 104406          GTSWR          ;;GET SOFT-SWR SETTINGS
004676 000403          BR 70$          ;;
004700 112737 000001 001134 69$: MOV #1,$AUTOB    ;;SET AUTO-MODE INDICATOR
      70$:
32 004706 005227 177777          INC #-1          ;;FIRST TIME THRU ?
33 004712 001002          BNE 1$          ;;BR IF NO
34 004714 104401 062040          TYPE ,PREAMB     ;;TYPE PRE-AMBLE MESSAGE
35 004720 005037 177776          CLR PS          ;;SET PROCESSOR STATUS TO 0
36 004724 012777 052740 174274 1$: MOV #RPVECT,@RPVEC ;;THIS IS FOR UNTIMELY DRIVE INTERRUPTS
37 004732 004737 054216          JSR PC,$TKINT   ;;INITIALIZE THE TTY KEYBOARD
38
39 004736 032777 010000 174174 RH70CK: BIT #SW12,@SWR   ;;LOOK TO SEE IF USING RH70
40 004744 001403          BEQ 1$          ;;IF SW12 = 0, SKIP NEXT
41 004746 012737 000001 001436      MOV #1,RH70     ;;IF SW12 = 1, CU IS AN RH70
42
43 004754 005737 001402          1$: TST SELECT     ;;200 START?
44 004760 001434          BEQ TST1        ;;SKIP NEXT IF STARTING FROM 200
45 004762 104401 004770          TYPE ,65$     ;;TYPE ASCIZ STRING
      004766 000422          BR 64$          ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ <CRLF>/SELECT UNIT NUMBER TO BE TESTED ?/
      64$:
46 005034 104412          RDOCT
47 005036 042716 177770          BIC #177770,(SP) ;;ONLY KEEP LAST 3 BITS
48 005042 011637 001374          MOV (SP),UNIT   ;;SAVE UNIT TO BE TESTED
49 005046 012637 001404          MOV (SP)+,UNITSL ;;SAVE UNIT TO BE TESTED
    
```

1
5

```

*****
*TEST 1 REFERENCE EACH REGISTER
*REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
*****
TST1: SCOPE
      MOV #1,$TIMES ;DO 1 ITERATION
      MOV #STACK,SP ;SET UP STACK POINTER
      MOV #1,TSTNM ;MOVE #1 TO TEST NUMBER
      MOV #REGSA1,EMTVEC ;ERROR VECTOR SO THAT NO REGISTERS ARE SAVED
      MOV ERRVEC,-(SP) ;PUSH ERRVEC ON STACK
      MOV #2$,ERRVEC ;SETUP FOR BUS TIMEOUT
      MOV #24,R0 ;THERE ARE 24 REG TO TEST
      MOV #RHDB,R1 ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
1$: MOV @(R1)+,R2 ;READ HARDWARE REG.
      DEC R0 ;COUNT DOWN
      BNE 1$ ;BRANCH IF 24 NOT DONE
      MOV (SP)+,ERRVEC ;POP STACK INTO ERRVEC
      BR 5$ ;BRANCH IF 24 DONE
20 005140 012716 005146 2$: MOV #3$, (SP) ;SETUP RETURN ADDRESS
21 005144 000002 RTI
22 005146 3$:
      MOV (SP)+,ERRVEC ;POP STACK INTO ERRVEC
      MOV -2(R1),$TMP1 ;STORE FAILING REG ADDR
      EMT 15
      BIT #SW13,@SWR ;INHIBIT ERROR PRINTOUT ?
      BNE 4$ ;BRANCH IF YES
      TYPE ,65$ ;TYPE ASCIZ STRING
      BR 64$ ;GET OVER THE ASCIZ
      ;:65$: .ASCIZ <CRLF>/TO CHANGE BASE ADDRESS, RESTART AT ADDRESS /
      64$:
28 005256 012746 000204 MOV #204,-(SP) ;GET READY TO TYPE STARTING ADDRESS
29 ;OF 'CHANGE OF BASE ADDRESS' ROUTINE
30 005262 104402 TYPOC
31 005264 000000 HALT ;FORCE THE RESTART!
32 005266 000137 040164 4$: JMP $EOP ;GO TO END OF PROGRAM
33
34 005272 5$:
      MOV ERRVEC,-(SP) ;PUSH ERRVEC ON STACK
      MOV #6$,ERRVEC ;INITIALIZE VECTOR
      CLR RH70 ;INIT RH INDICATOR ++ C.W
      TST @RHBAE ;ADDRESS RPBAE (RH11/RH70?)
      INC RH70 ;FOUND AN RH70-SET MASK
      TYPE ,67$ ;TYPE ASCIZ STRING
      BR 66$ ;GET OVER THE ASCIZ
      ;:67$: .ASCIZ <CRLF><LF>/RH70 CONTROLLER /
      66$:
40 005352 000420 BR 8$
41
42 005354 012716 005362 6$: MOV #7$, (SP) ;SETUP RETURN ADDRESS
43 005360 000002 RTI
44 005362 7$:
      TYPE ,69$ ;TYPE ASCIZ STRING
      BR 68$ ;GET OVER THE ASCIZ
      ;:69$: .ASCIZ <CRLF><LF>/RH11 CONTROLLER /
    
```

```

005414
45 005414 012737 055470 000030 68$:
46 005422 012637 000004 8$:      MOV      #ERROR,EMTVEC ;RESTORE ERROR VECTOR SO REGISTERS ARE SAVED
                                MOV      (SP)+,ERRVEC ;:POP STACK INTO ERRVEC
47
48 ;FIND THE SILO SIZE
49
50 005426 004737 041460      JSR      PC,CLDISK ;CONTROLLER CLEAR
51 005432 005037 001440      CLR      SILOSZ ;CLEAR SILO COUNTER
52 005436 013777 001440 173564 9$:      MOV      SILOSZ,@RHDB ;LOAD SILO
53 005444 005237 001440      INC      SILOSZ ;KEEP COUNT
54 005450 032777 000100 173560      BIT      #IR,@RHCS2 ;IS THE SILO FULL?
55 005456 001367          BNE      9$ ;BRANCH IF NO
56 005460 012737 000412 036150      MOV      #266.,VAR1+2 ;VAR1 IN TEST 116
57 005466 163737 001440 036150      SUB      SILOSZ,VAR1+2
58 005474 005437 036150      NEG      VAR1+2
59 005500 012737 001510 036156      MOV      #WRFROM,VAR2+2 ;VAR2 IN TEST 116
60 005506 063737 001440 036156      ADD      SILOSZ,VAR2+2
61 005514 063737 001440 036156      ADD      SILOSZ,VAR2+2
62 005522 062737 001000 036156      ADD      #512.,VAR2+2
63 005530 004737 041460      JSR      PC,CLDISK ;CONTROLLER CLEAR
64
69 ;*****
;*TEST 2 RHCS2-CONTROL AND STATUS 2
;*THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
;*OF THE NUMBER OF DRIVES PRESENT
;*****
005534 000004
70 005536 012737 000001 001212 TST2:  SCOPE
71 005544 012706 001100      MOV      #1,$TIMES ;:DO 1 ITERATION
72 005550 012737 000002 001432      MOV      #STACK,SP ;:RESET STACK
                                MOV      #2,TSTNM ;:MOVE #2 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

005556 005737 001436      TST      RH70 ;:TEST FLAG FOR RH70 CONTROLLER
005562 001402          BEQ      64$ ;:IF FLAG = 1, THIS TEST IS SKIPPED
005564 000137 005620          JMP      TST3 ;:JUMP TO NEXT TEST
005570          64$: ;:IF FLAG = 0, DO THIS TEST

73
74 005570 013737 007610 005610      MOV      PRCS2+12,UN
75 005576 013737 001236 005612      MOV      RHCS2,UN+2
76 005604 004537 041132          JSR      R5,BITST ;:TEST BITS IN REGISTER
77 005610 000000          UN:      .WORD 0 ;:ONLY THESE BITS TESTED FOR READ/WRITE
78 005612 000000          .WORD 0 ;:ADDRESS OF REG. BEING TESTED
79 005614 104001          EMT      1
80 005616 000207          RTS      PC ;:RETURN TO BLT3 ROUTINE
81
82 ;*****
;*TEST 3 PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT
;*****
005620 000004
83 005622 012737 000001 001212 TST3:  SCOPE
84 005630 012737 000003 001432      MOV      #1,$TIMES ;:DO 1 ITERATION
                                MOV      #3,TSTNM ;:MOVE #3 TO TEST NUMBER

85 005636 013701 001256          MOV      RHAS,R1 ;:R1 HAS ADDRESS OF RHAS
86 005642 012711 177777          MOV      #-1,@R1 ;:THIS CLEARS RHAS (SURPRISED!)
87 005646 011137 001126          MOV      @R1,$BDDAT ;:TEST DATA
88 005652 105737 001126          TSTB     $BDDAT
  
```



```

89 005656 001405          BEQ     TST4      :BRANCH IF GOOD
90 005660 005037 001124   CLR     $GDDAT   :GOOD DATA
91 005664 010137 041130   MOV     R1,REGADR :FAILING REG. RHAS
92 005670 104001          EMT     1        :
93                                     :BY WRITING ONES IN
94
95
:*****
:*TEST 4          TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
:*****
TST4:  SCOPE
      MOV     #1,$TIMES      ;;DO 1 ITERATION
      RESET   :START WITH AN INIT
      MOV     #4,TSTNM      :MOVE #4 TO TEST NUMBER
      JSR    PC,$TKINT      :INITILIZE TTY KEYBOARD
      BIT     #SW13,@SWR    :INHIBIT ERROR TYPEOUT?
      BNE    4$             :BRANCH IF YES
      TYPE   ,65$          ;;TYPE ASCIZ STRING
      BR     ,64$          ;;GET OVER THE ASCIZ
      .ASCIZ <CRLF>/LOOKING AT RHAS - DRIVES PRESENT/
64$:
4$:   MOV     RHAS,R1        :LOAD R1 WITH ADDR. OF RHAS
      MOV     RHCS2,R2      :LOAD R2 WITH ADDR. OF RHCS2
      CLR     @R2           :CLEAR RHCS2 (ADDRESS UNIT #0)
      MOV     #8.,R0        :INITIALIZE DRIVE COUNTER
      MOV     RHER1,R4      :LOAD R4 WITH ADDR. OF RHER1
1$:   MOV     #-1,@R4       :MOVE ERRORS INTO RHER1 OF UNIT ADDRESSED
      INC    @R2           :INCREMENT UNIT NO. (RHCS2)
      DEC    R0            :COUNT DOWN DRIVE COUNTER
      BNE    1$           :TEST AND DO NEXT UNIT IF 8 NOT DONE
111
112 006032 111137 001420   MOVVB  @R1,TOTALAT   :SAVE ALL RESULTING ATTENTION BITS
113                                     : (USED IN DRIVE CLEAR TEST)
114 006036 105037 001421   CLRB   TOTALAT+1    :CLEAR UPPER BYTE
115 006042 105711          TSTB   @R1          :TEST RHAS FOR ANY DRIVES PRESENT
116 006044 001402          BEQ     2$           :NONE RESPONDING - TYPE THE MESSAGE
117 006046 000137 006076   JMP    XE2          :SOME THERE - GO FILL 'UNITS' TABLE
118
119 006052 032777 020000 173060 2$:   BIT     #SW13,@SWR   :INHIBIT ERROR TYPE OUT?
120 006060 001402          BEQ     3$           :TYPE 'NO DRIVES' MESSAGE IF NO
121 006062 000137 006424   JMP    SELTST      :CHECK FOR SELECTED UNIT START AND LOAD
122                                     : 'UNITS' TABLE WITH DESIRED DRIVE IF SO
123
124 006066 104401 062426   3$:   TYPE   ,NDRVAS    :TYPE 'NO DRIVES PRESENT IN RHAS'
125 006072 000137 040164   JMP    $EOP        :GO OUT
126
127                                     ;SET UP DRIVES PRESENT TABLE
128 006076   XE2:
129
130 006076 012700 000010   2$:   MOV     #8.,R0    :LOAD 'UNITS' TABLE COUNTER
131 006102 012703 0C1354   MOV     #UNITS,R3   :LOAD 'UNITS' TABLE POINTER
132 006106 012723 177777   3$:   MOV     #-1,(R3)+ :PRESET 1ST TABLE BLOCK TO ALL ONES
133 006112 005300          DEC    R0           :COUNT DOWN
134 006114 001374          BNE    3$          :PRESET NEXT BLOCK IF 8 NOT DONE
135
136 006116 012703 001354   10$:  MOV     #UNITS,R3   :RELOAD THE TABLE POINTER
137 006122 005005          CLR    R5          :INITIALIZE UNIT NO. TO 0
138 006124 005037 001376   CLR    NOUNIT ;NO. OF UNITS PRESENT
    
```

```

139 006130 012700 000010      MOV      #8,R0      ;RELOAD THE TABLE COUNTER
140 006134 011137 001176      MOV      @R1,$TMP0  ;ADDR OF RHAS INTO TEMPORARY STORAGE
141 006140 006037 001176      ROR      $TMP0      ;SET CARRY IF 0 BIT = 1 (UNIT ATTEN.)
142 006144 103114              BCC      5$         ;CHECK NEXT UNIT IF ONE NOT IN BIT 0
143
144 006146 010577 173064      MOV      R5,@RHCS2  ;INSERT UNIT NO. INTO RHCS2 UNIT ADDR.
145 006152 022777 024020 173104  CMP      #24020,@RHDT ;READ RHDT - IS IT A DUAL PORT RP04 ?
146 006160 001477              BEQ      6$         ;YES...TYPE THE UNIT NO.
147 006162 022777 020020 173074  CMP      #20020,@RHDT ;READ RHDT - IS IT A SINGLE PORT RP04 ?
148 006170 001473              BEQ      6$         ;YES...TYPE THE UNIT NO.
149
150 006172 022777 024021 173064  CMP      #24021,@RHDT ;DUAL PORT RP05 ?
151 006200 001467              BEQ      6$         ;TYPE UNIT NO. IF SO
152 006202 022777 020021 173054  CMP      #20021,@RHDT ;SINGLE PORT RP05 ?
153 006210 001463              BEQ      6$         ;TYPE NO. IF SO
154
155 006212 022777 024022 173044  CMP      #24022,@RHDT ;READ RHDT - IS IT A DUAL PORT RP06 ?
156 006220 001457              BEQ      6$         ;YES...TYPE THE UNIT NO.
157 006222 022777 020022 173034  CMP      #20022,@RHDT ;READ RHDT - IS IT A SINGLE PORT RP06 ?
158 006230 001453              BEQ      6$         ;YES...TYPE THE UNIT NO.
159
160 006232 104401 006240      TYPE     ,65$      ;;TYPE ASCIZ STRING
    006236 000407      BR       64$      ;;GET OVER THE ASCIZ
    ;;65$: .ASCIZ <CRLF>/UNIT NUMBER /
    64$
    006256
161 006256 010546      MOV      R5,-(SP)  ;PUT THE UNIT NUMBER ON STACK
162 006260 104405      TYPDS   ;TYPE IT
163 006262 104401 006270      TYPE     ,67$      ;;TYPE ASCIZ STRING
    006266 000405      BR       66$      ;;GET OVER THE ASCIZ
    ;;67$: .ASCIZ /, RHDT = /
    66$
164 006302 017746 172756      MOV      @RHDT,-(SP) ;PUT RHDT ON THE STACK
165 006306 104402      TYPOC   ;TYPE IT
166 006310 104401 006316      TYPE     ,69$      ;;TYPE ASCIZ STRING
    006314 000420      BR       68$      ;;GET OVER THE ASCIZ
    ;;69$: .ASCIZ ? - NOT AN RP04/RP05/RP06 DEVICE?
    68$
167 006356 000407      BR       5$         ;UNIT NOT AN RP04/RP05/RP06 SO TEST NEXT ONE
168
169 006360 010523      MOV      R5,(R3)+  ;LOAD TABLE POSITION AND INCR IT
170 006362 104401 001223      TYPE     ,CRLF     ;CRLF
171 006366 010546      MOV      R5,-(R5)  ;PUT UNIT NO. ON THE STACK
172 006370 104405      TYPDS   ;TYPE THE UNIT NO.
173 006372 005237 001376      INC      NUNIT     ;INCR THE TOTAL NO. OF UNITS
174
175 006376 005205      INC      R5        ;'RHCS2' UNIT ADDRESS
176 006400 005300      DEC      R0        ;DRIVE COUNTER DOWN ONE
177 006402 001256      BNE     4$         ;TEST AND DO NEXT UNIT IF 8 NOT DONE
178
179 006404 013737 001354 001374 12$: MOV      UNITS,UNIT ;SET UNIT NO. TO FIRST ONE FOUND/OR 0
180 006412 013737 001376 001400  MOV      NUNIT,NUNIT ;SAVE NO. OF UNITS
181 006420 005337 001400      DEC      NUNIT     ;IF NUNIT = 0 THEN ONLY ONE UNIT
182
183
184 006424 005737 001402      SELTST: TST      SELECT ;STARTING ADDRESS 200 ?
185 006430 001403      BEQ     TST5      ;BRANCH IF STARTING FROM 200
186 006432 013737 001404 001374  MOV      UNITSL,UNIT ;CHANGE UNIT NUMBER TO SELECTED ONE

```

187
197

```

:*****
:*TEST 5      TYPE SERIAL NUMBER AND DRIVE TYPE
:*SET APPROPRIATE ATTENTION BIT OF UNIT UNDER TEST IN
:*'ATTENT' AND TYPE THE UNIT UNDER TEST
:*
:*READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER,
:*TYPE THEM OUT AND PROCEED
:*
:*TO LOOP HERE SET SWITCH 8 AND THIS TEST NO. AND RESTART
:*****

```

006440 000004
 006442 012737 000001 001212
 006450 012737 007156 001106
 198 006456 012737 000005 001432
 199 006464 004737 041460
 200 006470 005037 001416

```

TST5:  SCOPE
      MOV  #1,$TIMES      ;;DO 1 ITERATION
      MOV  #1,$SLPADR    ;;SET SCOPE LOOP ADDRESS
      MOV  #5,$TSTNM     ;;MOVE #5 TO TEST NUMBER
      JSR  PC,CLDISK     ;;FILL UNIT NO.
      CLR  ATTENT        ;CLEAR

```

201
202
203
204 006474 005737 001374
205 006500 001022
206 006502 012700 000041
207 006506 122710 000011
208 006512 001015
209 006514 005737 001402

```

;TEST FOR UNIT #0
      TST  UNIT          ;IS UNIT #0 NEXT IN THE UNITS TABLE ?
      BNE  10$          ;IF NOT, TEST THIS UNIT
      MOV  #41,$RO      ;IF SO, CHECK THE LOAD MEDIA LOCATION
      CMPB #11,($RO)    ;WAS IT AN RP04/5/6 ?
      BNE  10$          ;NO...GO AHEAD WITH TESTING UNIT #0
      TST  SELECT       ;WAS UNIT #0 SELECTED ?
                        ;(IE. WAS IT A 210 START ?)
      BNE  10$          ;IF SO...TEST IT

```

210
211 006520 001012
212
213
214
215

```

;INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
;& DECREMENT THE 'NOUNITS' PRESENT (TO BE TESTED)

```

216 006522 012700 001354
217 006526 005720
218
219 006530 022710 177777
220 006534 001404
221 006536 011037 001374
222 006542 005337 001376
223 006546 013700 001374

```

      MOV  #UNITS,$RO   ;LOAD THE UNITS TABLE POINTER
      TST  ($RO)+       ;SELECT THE NEXT UNIT IN THE TABLE
                        ;(DOUBLE INCREMENT THE POINTER, RO)
      CMP  #-1,($RO)    ;IS THERE ANOTHER TABLE ENTRY PRESENT ?
      BEQ  10$          ;IF NOT (LOC = -1)...MUST USE UNIT #0
      MOV  ($RO),UNIT   ;SET UP TO BE THE UNIT UNDER TEST
      DEC  NOUNITS      ;DECREMENT BECAUSE UNIT # 0 WON'T BE TESTED
10$:  MOV  UNIT,$RO     ;RO CONTAINS UNIT NO.

```

224
225
226

```

;SET UP THE PROPER DEVICE TYPE FLAG

```

227 006552 010077 172460
228 006556 005037 001434
229 006562 022777 024022 172474
230 006570 001405
231 006572 022777 020022 172464
232 006600 001401
233 006602 000403
234 006604 012737 177777 001434

```

      MOV  $RO,@RHCS2   ;SET UP UNIT ADDRESS
      CLR  RP06         ;CLEAR RP06 DEVICE TYPE FLAG
      CMP  #24022,@RHDT ;DUAL PORT RP06 ?
      BEQ  2$          ;YES...SET THE FLAG
      CMP  #20022,@RHDT ;SINGLE PORT RP06 ?
      BEQ  2$          ;YES..SET FLAG
      BR   3$          ;NO...DON'T SET RP06 FLAG
2$:  MOV  #-1,$RP06    ;SET IT

```

235
236
237

```

;ASSUME THE NEXT UNIT IS AN RP04

```

238 006612 116037 003620 001416
239 006620 104401 006626
006624 000413

```

3$:  MOVB  ATABLE($RO),ATTENT ;SET APPROPRIATE ATTENTION BIT
      TYPE ,65$              ;;TYPE ASCIZ STRING
      BR   64$              ;;GET OVER THE ASCIZ

```

```

    006654      64$: .ASCIZ <CRLF>/TESTING DRIVE NUMBER/
240 006654 013746 001374      MOV UNIT,-(SP)      :UNIT NO. TO STACK
241 006660 104405      TYPDS      :TYPE DRIVE NO.
242 006662 104401 006670      TYPE      ,67$      :TYPE ASCIZ STRING
    006666 000410      BR      66$      :GET OVER THE ASCIZ
    67$: .ASCIZ <CRLF>/SERIAL NO. = /
    66$:
243 006710 017746 172352      MOV @RHSN,-(SP)    :SAVE @RHSN FOR TYPEOUT
    006714 104402      TYPOC      :GO TYPE--OCTAL ASCII(ALL DIGITS)
244 006716 104401 006724      TYPE      ,69$      :TYPE ASCIZ STRING
    006722 000410      BR      68$      :GET OVER THE ASCIZ
    69$: .ASCIZ <CRLF>/DRIVE TYPE = /
    68$:
245 006744 017746 172314      MOV @RHDT,-(SP)    :SAVE @RHDT FOR TYPEOUT
    006750 104402      TYPOC      :GO TYPE--OCTAL ASCII(ALL DIGITS)
246
247 006752 022777 024020 172304      CMP #24020,@RHDT   :DUAL PORT RPO4 ?
248 006760 001425      BEQ 4$           :TYPE ASCII MSG OUT
249 006762 022777 020020 172274      CMP #20020,@RHDT   :SINGLE PORT RPO4 ?
250 006770 001421      BEQ 4$           :TYPE THE MESSAGE
251
252 006772 022777 024021 172264      CMP #24021,@RHDT   :DUAL PORT RPO5 ?
253 007000 001451      BEQ 6$           :TYPE MSG
254 007002 022777 020021 172254      CMP #20021,@RHDT   :SINGLE PORT RPO5 ?
255 007010 001445      BEQ 6$           :TYPE MSG
256
257 007012 022777 024022 172244      CMP #24022,@RHDT   :DUAL PORT RPO6 ?
258 007020 001423      BEQ 5$           :TYPE MSG
259 007022 022777 020022 172234      CMP #20022,@RHDT   :SINGLE PORT RPO6 ?
260 007030 001417      BEQ 5$           :TYPE MSG
261 007032 000451      BR 1$           :DRIVE IS NOT AN RPO4/RPO5/RPO6 - SO
262      :DON'T TYPE THE ASCII MESSAGE
263      :SHOULD NEVER HAPPEN AT THIS POINT
264      :UNLESS DRIVE GOT SICK WHILE TESTING
265      :WAS IN PROGRESS
266
267 007034      4$:
    007034 104401 007042      TYPE      ,71$      :TYPE ASCIZ STRING
    007040 000412      BR      70$      :GET OVER THE ASCIZ
    71$: .ASCIZ <CRLF>/DRIVE IS AN RPO4/<CRLF>
    70$:
268 007066 000433      BR 1$           :SKIP NEXT MESSAGE
269 007070      5$:
    007070 104401 007076      TYPE      ,73$      :TYPE ASCIZ STRING
    007074 000412      BR      72$      :GET OVER THE ASCIZ
    73$: .ASCIZ <CRLF>/DRIVE IS AN RPO6/<CRLF>
    72$:
270 007122 000415      BR 1$           :SKIP NEXT
271 007124      6$:
    007124 104401 007132      TYPE      ,75$      :TYPE ASCIZ STRING
    007130 000412      BR      74$      :GET OVER THE ASCIZ
    75$: .ASCIZ <CRLF>/DRIVE IS AN RPO5/<CRLF>
    74$:
272
273 007156 005777 172104      1$: TST @RHSN      :READ SERIAL NO. AND DRIVE TYPE
274 007162 005777 172076      TST @RHDT      :THESE TWO ARE TO HELP SCOPE LOOPS
  
```

```

275 007166 017737 172074 001412      MOV  @RHSN,SAVSN      ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
276 007174 017737 172064 001410      MOV  @RHDT,SAVDT     ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
277
283
  
```

```

:*****
:*TEST 6      CHECK MOL TO BE LOW
:*MAKE SURE THAT DRIVE IS OFF LINE BEFORE STARTING PROGRAM
:*IF DRIVE IS ON LINE THEN AFTER TYPE OUT THE PROGRAM WILL
:*HANG FOR EVER WAITING FOR DRIVE TO GO OFF LINE
:*****
  
```

```

007202 000004
284 007204 012737 000006 001432      TST6:  SCOPE
285 007212 004737 041460              MOV  #6,TSTNM        ;MOVE #6 TO TEST NUMBER
286 007216 032713 010000              JSR  PC,CLDISK      ;GIVE INITILIZE
287 007222 001542                      BIT  #MOL,@R3       ;CHECK MOL IN RHDS1
288 007224 104401 007232              BEQ  TST7           ;BRANCH IF MOL LOW
007230 000420                      TYPE ,65$          ;;TYPE ASCIZ STRING
                                BR   64$            ;;GET OVER THE ASCIZ
                                ;;65$: .ASCIZ <CRLF>/DRIVE IS ON LINE - MOL IS HIGH/
                                64$:
289 007272 104401 007300              TYPE ,67$          ;;TYPE ASCIZ STRING
007276 000423              BR   66$            ;;GET OVER THE ASCIZ
                                ;;67$: .ASCIZ <CRLF>/HIT STOP ON DRIVE TO GET IT OFF LINE/
                                66$:
290 007346 104401 007354              TYPE ,69$          ;;TYPE ASCIZ STRING
007352 000430              BR   68$            ;;GET OVER THE ASCIZ
                                ;;59$: .ASCIZ <CRLF>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
                                68$:
291 007434 032713 010000              1$:  BIT  #MOL,@R3   ;CHECK MOL IN RHDS1
292 007440 001375                      BNE  1$            ;BRANCH IF MOL IS HIGH
293 007442 104401 007450              TYPE ,71$          ;;TYPE ASCIZ STRING
007446 000430              BR   70$            ;;GET OVER THE ASCIZ
                                ;;71$: .ASCIZ <CRLF><LF>/MOL IS NOW LOW. PROGRAM WILL NOW BE EXECUTED/<CRLF>
                                70$:
007530
  
```

```
1  
2  
8  
          .SBTTL REGISTER TESTS  
          :*****  
          :*TEST 7      RHCS2 - CONTROL AND STATUS 2  
          :*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
          :*REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
          :*WALKING 1'S (1,2,4,10 ETC)  
          :*****  
9 007530 000004  
10 007532 012737 000007 001432  TST7:  SCOPE  
          MOV      #7,TSTNM      ;MOVE #7 TO TEST NUMBER  
          :CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70  
          :  
          007540 005737 001436      TST      RH70      ;TEST FLAG FOR RH70 CONTROLLER  
          007544 001402      BEQ      64$      ;IF FLAG = 1, THIS TEST IS SKIPPED  
          007546 000137 007620      JMP      TST10     ;JUMP TO NEXT TEST  
          007552      64$:      ;IF FLAG = 0, DO THIS TEST  
11 007552 004737 041460      JSR      PC,CLDISK ;GIVE INITIALIZE  
12 007556 012706 001100      MOV      #STACK,SP ;RESET STACK  
          007562 012737 000007 001432  MOV      #7,TSTNM  ;MOVE #7 TO TEST NUMBER  
          :  
          007570 013737 001236 007612  PRCS2:  MOV      RHCS2,PRCS2+14 ;GET REGISTER ADDRESS  
          007576 013777 001374 171432  MOV      UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST  
          007604 004537 041132      JSR      R5,BITST  ;TEST BITS IN REGISTER  
          007610 020017      .WORD    20017      ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
          007612 176710      .WORD    176710     ;ADDRESS OF REG. BEING TESTED  
          007614 104001      EMT      1  
          007616 000207      RTS      PC      ;RETURN TO BLT3 ROUTINE  
13  
19  
          :*****  
          :*TEST 10     RHCS1 - CONTROL AND STATUS 1 REGISTER  
          :*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
          :*REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
          :*WALKING 1'S (1,2,4,10 ETC)  
          :*****  
20 007620 000004  
          007622 012706 001100  
          007626 012737 000010 001432  TST10:  SCOPE  
          MOV      #STACK,SP      ;RESET STACK  
          MOV      #10,TSTNM     ;MOVE #10 TO TEST NUMBER  
          :  
          007634 013737 001240 007656  PRCS1:  MOV      RHCS1,PRCS1+14 ;GET REGISTER ADDRESS  
          007642 013777 001374 171366  MOV      UNIT,@RHCS1 ;MOVE UNIT NO. UNDER TEST  
          007650 004537 041132      JSR      R5,BITST  ;TEST BITS IN REGISTER  
          007654 001476      .WORD    1476      ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
          007656 176700      .WORD    176700     ;ADDRESS OF REG. BEING TESTED  
          007660 104001      EMT      1  
          007662 000207      RTS      PC      ;RETURN TO BLT3 ROUTINE  
21  
27  
          :*****  
          :*TEST 11     RHCS1 - BIT # 13 - MCPE  
          :*THIS FORCES A MASS BUS CONTROL PARITY ERROR  
          :*BY SETTING 'PAT', LOOKING FOR 'PAR', WRITING RHCS1  
          :*AND READING RHER1  
          :*****  
28 007664 000004  
          007666 012706 001100  TST11:  SCOPE  
          MOV      #STACK,SP      ;RESET STACK
```

```
29 007672 012737 000011 001432      MOV    #11,TSTNM      ;MOVE #11 TO TEST NUMBER
30 007700 004737 041460              JSR    PC,CLDISK      ;INIT AND SET UNIT NUMBER AND DEVICE -
31                                     ;CPU REG. CORRESPONDENCE (R1-R4)
32
33                                     ;SET FORCED PARITY ERROR 'PAT'
34 007704 052777 000020 171324      BIS    #PAT,@RHCS2    ;SET 'PAT' TO INVERT PARITY
35                                     ;GENERATED
36 007712 005077 171324              CLR    @RHER1         ;WRITE DCL REGISTER USING BAD CONTROL PARITY
37
38                                     ;WITH THIS PARITY ERROR NOTHING WILL BE READ TILL IT IS
39                                     ;CLEARED
40
41 007716 011137 001126              MOV    @R1,$BDDAT     ;RHCS1 $BDDAT
42 007722 022737 104200 001126      CMP    #SC!DVA!RDY,$BDDAT ;COMPARE RHCS1 AFTER PARITY
43                                     ;ERROR
44 007730 001406                    BEQ    1$              ;BRANCH IF SC!DVA!RDY=1
45 007732 012737 104200 001124      MOV    #SC!DVA!RDY,$GDDAT ;GOOD DATA
46 007740 010137 041130              MOV    R1,REGADR      ;REGISTER ADDRESS RHCS1
47 007744 104001                    EMT    1
48                                     ;WRITING DCL REGISTER RHCS1
49                                     ;DID NOT SET SC!DVA!RDY
50                                     ;WITH 'PAT' BIT HIGH
51 007746 013746 001374 1$:        MOV    UNIT,-(SP)      ;GET UNIT NUMBER
52 007752 052716 000120              BIS    #PAT!IR,(SP)   ;INCLUDE PAT AND IR
53 007756 012637 001124              MOV    (SP)+,$GDDAT   ;PUT ON STACK
54 007762 011237 001126              MOV    @R2,$BDDAT     ;RHCS2 $BDDAT
55 007766 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE RHCS2
56 007774 001403                    BEQ    2$              ;OK - SC!DVA!RDY ARE HIGH
57 007776 010237 041130              MOV    R2,REGADR      ;REGISTER ADDRESS
58 010002 104001                    EMT    1
59                                     ;SHOW UNIT#!PAT!IR BITS HIGH
60
61 010004 011437 001126 2$:        MOV    @R4,$BDDAT     ;RHER1 $BDDAT
62 010010 022737 000010 001126      CMP    #PAR,$BDDAT   ;ERROR REGISTER RHER1 SHOULD
63                                     ;HAVE 'PAR' SET
64 010016 001406                    BEQ    3$              ;A - OK, IT DOES
65 010020 012737 000010 001124      MOV    #PAR,$GDDAT    ;GOOD DATA
66 010026 010437 041130              MOV    R4,REGADR      ;FAILING REGISTER RHER1
67 010032 104001                    EMT    1
68                                     ;SET 'PAR'
69 010034 3$:
70
76
```

```
::*****
:*TEST 12      RHWC - WORD COUNT REGISTER
:*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
:*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
:*WALKING 1'S (1,2,4,10 ETC)
:*****
```

```
77 010034 000004      TST12: SCOPE
010036 012706      MOV    #STACK,SP      ;RESET STACK
010042 012737 000012 001432      MOV    #12,TSTNM     ;MOVE #12 TO TEST NUMBER

010050 013737 001232 010072      PRWC: MOV    RHWC,PRWC+14 ;GET REGISTER ADDRESS
010056 013777 001374 171152      MOV    UNIT,@RHCS2   ;MOVE UNIT NO. UNDER TEST
010064 004537 041132              JSR    R5,BITST      ;TEST BITS IN REGISTER
010070 177777              .WORD 177777         ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
010072 176702              .WORD 176702         ;ADDRESS OF REG. BEING TESTED
```



```

010074 104001          EMT 1
010076 000207          RTS PC          ;RETURN TO BLT3 ROUTINE

78
84
*****
*TEST 13      RHBA - UNIBUS ADDRESS REGISTER
*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
*WALKING 1'S (1,2,4,10 ETC)
*****
85 010100 000004
010102 012706 001100
010106 012737 000013 001432
      TS113: SCOPE
            MOV #STACK,SP          ;RESET STACK
            MOV #13,TSTNM          ;MOVE #13 TO TEST NUMBER

010114 013737 001234 010136
010122 013777 001374 171106 PRBA: MOV RHBA,PRBA+14 ;GET REGISTER ADDRESS
            MOV UNIT,@RHCS2        ;MOVE UNIT NO. UNDER TEST
            JSR R5,BITST           ;TEST BITS IN REGISTER
            .WORD 177776           ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
            .WORD 176704           ;ADDRESS OF REG. BEING TESTED
010134 177776
010136 176704
010140 104001          EMT 1
010142 000207          RTS PC          ;RETURN TO BLT3 ROUTINE

86
92
*****
*TEST 14      RHER1 - ERROR REGISTER #1
*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
*WALKING 1'S (1,2,4,10 ETC)
*****
93 010144 000004
010146 012706 001100
010152 012737 000014 001432
      TST14: SCOPE
            MOV #STACK,SP          ;RESET STACK
            MOV #14,TSTNM          ;MOVE #14 TO TEST NUMBER

010160 013737 001242 010202
010166 013777 001374 171042 PRER1: MOV RHER1,PRER1+14 ;GET REGISTER ADDRESS
            MOV UNIT,@RHCS2        ;MOVE UNIT NO. UNDER TEST
            JSR R5,BITST           ;TEST BITS IN REGISTER
            .WORD 177777           ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
            .WORD 176714           ;ADDRESS OF REG. BEING TESTED
010200 177777
010202 176714
010204 104001          EMT 1
010206 000207          RTS PC          ;RETURN TO BLT3 ROUTINE

94
100
*****
*TEST 15      RHMR - MAINTENANCE REGISTER
*BIT 0 (DMD) MUST BE SET BEFORE THE OTHER BITS
*ARE READ WRITE
*ONLY 5 LOW ORDER BITS ARE TESTED (R2 HAS 5)
*****
101 010210 000004
102 010212 012737 000015 001432
103 010220 004737 041460
104 010224 013700 001260
105 010230 012701 000001
106 010234 012702 000005
107 010240 012710 000001
108 010246 010146
      TST15: SCOPE
            MOV #15,TSTNM          ;MOVE #15 TO TEST NUMBER
            JSR PC,CLDISK          ;SET UNIT NUMBER AND INIT
            MOV RHMR,R0 ;R0 HAS MAINTENANCE REG. ADR.
            MOV #1,R1             ;R1 HAS DATA
            MOV #5,R2             ;R2 HAS COUNT OF NUMBER OF BITS
1$: MOV #DMD,@R0                 ;SET DIAGNOSTIC MODE BIT
            BIS R1,@R0            ;SET DATA IN RHMR
            MOV R1,-(SP)          ;SAVE DATA FOR COMPARES
  
```

```

109 010250 052716 000401      BIS      #DMD!400,(SP)  ;INCLUDE BIT 0
110 010254 011637 001124      MOV      (SP), $GDDAT ;SAVE FOR ERROR PRINTOUT
111 010260 022610                CMP      (SP)+, @RO   ;COMPARE DATA
112 010262 001405                BEQ      2$           ;BRANCH IF GOOD
113 010264 011037 001126      MOV      @RO, $BDDAT  ;BAD DATA
114 010270 010037 041130      MOV      RO, REGADR   ;FAILING REG. ADR.
115 010274 104001                EMT      1
116                                ;FAILED TO SET INDICATED
117                                ;BITS
118 010276 000241                2$:     CLC           ;CLEAR CARRY
119 010300 006101                ROL      R1           ;GET NEXT DATA
120 010302 052701 000400      BIS      #400, R1     ;SET UNUSED BITS
121 010306 042701 001000      BIC      #BIT09, R1   ;CLEAR READ ONLY BIT
122 010312 005302                DEC      R2           ;COUNT
123 010314 001351                BNE      1$           ;BRANCH IF 5 BITS NOT DONE
124
125                                ;NOW FLOAT A 0
126
127
128 010316 012701 000435      MOV      #435, R1     ;R1 HAS DATA
129 010322 012702 000005      MOV      #5, R2      ;R2 HAS COUNT BITS
130 010326 012710 000001      3$:     MOV      #DMD, @RO ;SET DIAGNOSTIC MODE BITS
131 010332 050110                BIS      R1, @RO     ;SET DATA IN RHMR
132 010334 020110                CMP      R1, @RO     ;COMPARE DATA
133 010336 001407                BEQ      4$           ;BRANCH IF GOOD
134 010340 010137 001124      MOV      R1, $GDDAT  ;GOOD DATA
135 010344 011037 001126      MOV      @RO, $BDDAT ;BAD DATA
136 010350 010037 041130      MOV      RO, REGADR  ;FAILING REG. ADR. RHMR
137 010354 104001                EMT      1
138                                ;DOES NOT ALLOW WRITING
139                                ;ZEROS
140 010356 000261                4$:     SEC           ;SET CARRY
141 010360 006101                ROL      R1           ;GET NEXT DATA
142 010362 042701 001340      BIC      #BIT05!BIT06!BIT07!BIT09, R1 ;CLEAR READ ONLY BIT
143 010366 052701 000400      BIS      #BIT08, R1  ;SET BIT ZEROED BY ROL
144 010372 005302                DEC      R2           ;COUNT IF 5 BITS DONE
145 010374 001354                BNE      3$           ;BRANCH IF INCOMPLETE
146
147
148
149
150
151
152
*****
;*TEST 16      RHDST - DESIRED SECTOR/TRACK ADDRESS
;*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
;*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
;*WALKING 1'S (1,2,4,10 ETC)
*****
153 010376 000004                TST16:  SCOPE
010400 012706 001100                MOV      #STACK, SP  ;RESET STACK
010404 012737 000016 001432      MOV      #16, TSTNM  ;MOVE #16 TO TEST NUMBER

010412 013737 001244 010434      PRDST:  MOV      RHDST, PRDST+14 ;GET REGISTER ADDRESS
010420 013777 001374 170610      MOV      UNIT, @RHCS2 ;MOVE UNIT NO. UNDER TEST
010426 004537 041132                JSR      R5, BITST   ;TEST BITS IN REGISTER
010432 017437                .WORD   17437        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
010434 176706                .WORD   176706        ;ADDRESS OF REG. BEING TESTED
010436 104001                EMT      1
010440 000207                RTS      PC           ;RETURN TO BLT3 ROUTINE

```

```

160
:*****
:*TEST 17      RHER2 - ERROR REGISTER #2
:*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
:*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
:*WALKING 1'S (1,2,4,10 ETC)
:*****
161 010442 000004
010444 012706 001100
010450 012737 000017 001432
TST17: SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #17,TSTNM      ;MOVE #17 TO TEST NUMBER

010456 013737 001246 010500
010464 013777 001374 170544 PRER2: MOV      RHER2,PRER2+14 ;GET REGISTER ADDRESS
        MOV      UNIT,@RHCS2    ;MOVE UNIT NO. UNDER TEST
        JSR      R5,BITST      ;TEST BITS IN REGISTER
        .WORD    177777        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
        .WORD    176740        ;ADDRESS OF REG. BEING TESTED
010472 004537 041132
010476 177777
010500 176740
010502 104001
010504 000207
        EMT      1
        RTS      PC            ;RETURN TO BLT3 ROUTINE
  
```

```

162
168
:*****
:*TEST 20      RHOF - MARGIN/OFFSET REGISTER
:*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
:*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
:*WALKING 1'S (1,2,4,10 ETC)
:*****
169 010506 000004
010510 012706 001100
010514 012737 000020 001432
TST20: SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #20,TSTNM      ;MOVE #20 TO TEST NUMBER

010522 013737 001250 010544
010530 013777 001374 170500 PROF: MOV      RHOF,PROF+14 ;GET REGISTER ADDRESS
        MOV      UNIT,@RHCS2    ;MOVE UNIT NO. UNDER TEST
        JSR      R5,BITST      ;TEST BITS IN REGISTER
        .WORD    16277        ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
        .WORD    176732        ;ADDRESS OF REG. BEING TESTED
010536 004537 041132
010542 016277
010544 176732
010546 104001
010550 000207
        EMT      1
        RTS      PC            ;RETURN TO BLT3 ROUTINE
  
```

```

170
176
:*****
:*TEST 21      RHCA - DESIRED CYLINDER REGISTER
:*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE
:*REGISTERS.  USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND
:*WALKING 1'S (1,2,4,10 ETC)
:*****
177 010552 000004
010554 012706 001100
010560 012737 000021 001432
TST21: SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #21,TSTNM      ;MOVE #21 TO TEST NUMBER

010566 013737 001252 010610
010574 013777 001374 170434 PRCA: MOV      RHCA,PRCA+14 ;GET REGISTER ADDRESS
        MOV      UNIT,@RHCS2    ;MOVE UNIT NO. UNDER TEST
        JSR      R5,BITST      ;TEST BITS IN REGISTER
        .WORD    1777          ;ONLY THESE BITS ARE TESTED FOR READ/WRITE
        .WORD    176734        ;ADDRESS OF REG. BEING TESTED
010602 004537 041132
010606 001777
010610 176734
010612 104001
010614 000207
        EMT      1
        RTS      PC            ;RETURN TO BLT3 ROUTINE
  
```

178

184

```
*****  
*TEST 22 RHER3 - ERROR REGISTER #3  
*TEST LOADING AND READING OF ALL POSSIBLE BITS IN THE HARDWARE  
*REGISTERS. USE A PATTERN OF WALKING 0'S (-2,-3,-5 ETC.) AND  
*WALKING 1'S (1,2,4,10 ETC)  
*****
```

185 010616 000004
010620 012706 001100
010624 012737 000022 001432

```
TST22: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #22,TSTNM ;MOVE #22 TO TEST NUMBER  
  
PRER3: MOV RHER3,PRER3+14 ;GET REGISTER ADDRESS  
MOV UNIT,@RHCS2 ;MOVE UNIT NO. UNDER TEST  
JSR R5,BITST ;TEST BITS IN REGISTER  
.WORD 177777 ;ONLY THESE BITS ARE TESTED FOR READ/WRITE  
.WORD 176742 ;ADDRESS OF REG. BEING TESTED  
EMT 1  
RTS PC ;RETURN TO BLT3 ROUTINE
```

010632 013737 001254 010654
010640 013777 001374 170370
010646 004537 041132
010652 177777
010654 176742
010656 104001
010660 000207

186
187
188
189
190
191
192
193
194
201

```
*****  
:OF THE TWENTY REGISTERS (4 IN RH11, 16 IN RP04) ONLY 12 ARE  
:CHECKED IN THE ABOVE TESTS  
:TWO ARE ALREADY TESTED (SERIAL NO. AND DRIVE TYPE)  
:THE OTHER 7 WHICH ARE RHDS1, RHLA, RHCC, RHEC1, RHEC1, RHEC2  
:ARE READ ONLY REGISTERS. ONE OR ZERO CANNOT BE WRITTEN  
:*****
```

```
*****  
*TEST 23 CONTROL AND STATUS 2 (RHCS 2) - 'NED'  
*THIS TESTS THE UNIT SELECT BITS #0-2 (US1-4)  
*AND NON-EXISTENT DRIVE BIT #12 (NED)
```

*THE OTHER RHCS2 BITS ARE NOT TESTED HERE

202 010662 000004
203 010664 012706 001100
204 010670 012737 000023 001432

```
TST23: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #23,TSTNM ;MOVE #23 TO TEST NUMBER  
  
JSR PC,CLDISK ;HERE IT IS USED TO SETUP HARDWARE/  
;CPU REGISTER CORRESPONDENCE  
;R1=RHCS1  
;R2=RHCS2  
;R3=RHDS1  
;R4=RHER1  
CLR ERFLG$ ;CLEAR ERROR FLAG
```

205 010676 004737 041460
206
207
208
209
210
211 010702 005037 001406
212

```
;SIMULATED DISK AREA WILL BE USED AS A TEMPORARY  
;STORAGE TABLE FOR DRIVES PRESENT DETERMINED FROM 'NED' = 0 IN RHCS2
```

213
214
215
216 010706 012701 050224
217 010712 012700 000010
218 010716 012721 177777
219 010722 005300
220 010724 001374
221
222 010726 005012
223 010730 012700 000010

```
1$: MOV #DISK,R1 ;LOAD TABLE POINTER  
MOV #8,R0 ;LOAD TABLE LOCATION COUNTER  
MOV #-1,(R1)+ ;FILL 8 LOCATIONS WITH -1  
DEC R0 ;COUNT DOWN ONE LOCATION  
BNE 1$ ;BRANCH IF 8 NOT DONE  
  
CLR @R2 ;SELECT UNIT NO.0 (U2!U1!U0=0)  
MOV #8,R0 ;RELOAD TABLE LOCATION COUNTER
```

```

224 010734 012701 050224          MOV    #DISK,R1      ;RELOAD THE TABLE POINTER
225 010740 005714          2$:   TST    @R4      ;READ A DRIVE REGISTER (RHER1)
226 010742 032712 010000          BIT    #NED,@R2    ;NON EXISTENT DRIVE BIT = 0 ?
227 010746 001415          BEQ    3$          ;YES...DRIVE PRESENT, CHECK THE TYPE
228 010750 005300          7$:   DEC    R0      ;NO...DECREMENT DRIVE COUNT
229 010752 001454          BEQ    4$          ;CHECK RESULTS IF 8 DRIVES DONE
230
231 010754 011246          10$:  MOV    @R2,-(SP)    ;PUT RHCS2 ON THE STACK
232 010756 042716 177770          BIC    #^C?,(SP)   ;MASK ALL BUT THE UNIT NUMBER
233 010762 005216          INC    (SP)        ;INCREMENT THE UNIT NUMBER
234 010764 013703 001240          MOV    RHCS1,R3    ;GET RHCS1 ADDRESS
235 010770 005203          INC    R3          ;ADDRESS UPPER BYTE OF RHCS1
236 010772 112713 000100          MOVB   #100,@R3    ;SET 'TRE' IN RHCS1
237                                ;WITHOUT ADDRESSING DRIVE
238 010776 012612          MOV    (SP)+,@R2   ;RHCS2 HAS THE INCREMENTED UNIT
239                                ;WITH 'NED' CLEARED
240 011000 000757          BR     2$          ;TEST FOR NEXT DRIVE
241
242                                ;CHECK THE UNIT TYPE AND BUILD 'NED' DERIVED UNITS TABLE
243
244 011002 022777 024020 170254 3$:   CMP    #24020,@RHDT ;IS THIS A DUAL PORT RP04 ?
245 011010 001425          BEQ    8$          ;ENTER IN TABLE IF SO
246 011012 022777 020020 170244  CMP    #20020,@RHDT ;IS THIS A SINGLE PORT RP04 ?
247 011020 001421          BEQ    8$          ;ENTER IN TABLE IF SO
248
249 011022 022777 024022 170234  CMP    #24022,@RHDT ;IS THIS A DUAL PORT RP06 ?
250 011030 001415          BEQ    8$          ;ENTER IN TABLE IF SO
251 011032 022777 020022 170224  CMP    #20022,@RHDT ;IS THIS A SINGLE PORT RP06 ?
252 011040 001411          BEQ    8$          ;ENTER IN TABLE IF SO
253
254 011042 022777 024021 170214  CMP    #24021,@RHDT ;IS THIS A DUAL PORT RP05 ?
255 011050 001405          BEQ    8$          ;ENTER IN TABLE IF SO
256 011052 022777 020021 170204  CMP    #20021,@RHDT ;IS THIS A SINGLE PORT RP05 ?
257 011060 001401          BEQ    8$          ;ENTER IN TABLE IF SO
258 011062 000732          BR     7$          ;NO RP04 FOUND SO CHECK NEXT UNIT
259
260 011064 012746 000010          8$:   MOV    #8,-(SP)    ;LOAD MAX NO. OF DRIVES
261 011070 160016          SUB    R0,(SP)     ;(SP) NOW HAS THE PRESENT DRIVE NO.
262 011072 012621          MOV    (SP)+,(R1)+ ;LOAD TABLE, INCR TABLE LOCATION &
263                                ;RESTORE THE STACK TO WHERE IT WAS
264 011074 005300          DEC    R0          ;DECREMENT THE DRIVE COUNT
265 011076 001402          BEQ    4$          ;CHECK RESULTS IF 8 UNITS CHECKED
266 011100 005212          INC    @R2        ;SELECT NEXT UNIT
267 011102 000716          BR     2$          ;GO TEST IT
268
269                                ;COMPARE 'NED' DERIVED UNITS TABLE WITH THAT DERIVED USING RHAS IN T4
270
271 011104 004037 042354          4$:   JSR    R0,COMPAR   ;COMPARE RESULTS
272 011110 001354          UNITS              ;RHER1/RHAS DERIVED DATA
273 011112 050224          DISK              ;'NED' TEST DATA
274 011114 000010          8$              ;NO.OF WORDS TO COMPARE
275 011116 011124          5$              ;RETURN FOR ERROR HEADER
276 011120 011152          6$              ;RETURN FOR ERROR DATA
277 011122 011270          13$             ;RETURN FOR GOOD COMPARISON (NEXT TEST)
278
279                                ;SPECIAL 'NED'/'RHAS' TABLE TYPE OUT ROUTINE (BYPASSES .SERRTYP AND
280                                ;HENCE IGNORES INHIBIT ERROR TYPEOUT SWITCH)

```

```

281
282 011124          5$:      EMT      22
      011124 104022          MOV      #8.,R3      ;LENGTH OF BOTH UNIT TABLES
283 011126 012703 000010          MOV      #UNITS,R1  ;ADDRESS G. RHAS/RHER1 UNITS TABLE
284 011132 012701 001354          MOV      #DISK,R2   ;ADDRESS OF 'NED' RHCS2 UNITS TABLE
285 011136 012702 050224          MOV      (R1)+,$GDDAT ;LOAD RHAS UNIT NO.INTO '$GDDAT' AND
286 011142 012137 001124          14$:    MOV      ;INCREMENT THE TABLE LOCATION
287
288 011146 012237 001126          MOV      (R2)+,$BDDAT ;LOAD 'NED' UNIT NO. INTO '$BDDAT'
289
290
291 011152 032777 020000 167760 6$:      BIT      #SW13,@SWR   ;INHIBIT ERROR TYPE OUTS ?
292 011160 001043          BNE      13$        ;YES...EXIT
293 011162 022737 177777 001124          CMP      #-1,$GDDAT ;DOES RHAS UNIT TABLE LOCATION = -1 ?
294 011170 001413          BEQ      11$        ;YES...DON'T TYPE IT - CHECK 'NED' TABLE
295 011172 104401 063174          TYPE    ,BLNKS8    ;NO...TAB OVER PC COLUMN
296 011176 104401 063174          TYPE    ,BLNKS8    ;TAB OVER THE TEST NO. COLUMN
297 011202 013746 001124          MOV      $GDDAT,-(SP) ;SAVE $GDDAT FOR TYPEOUT
      011206 104403          TYPOS    ;GO TYPE--OCTAL ASCII
      011210      006          .BYTE    6          ;TYPE 6 DIGITS
      011211      000          .BYTE    0          ;SUPPRESS LEADING ZEROS
298 011212 104401 063202          TYPE    ,BLNKS2    ;SPACE OVER TO THE NEXT COLUMN
299 011216 000406          BR       12$      ;CHECK THE 'NED' UNIT TABLE
300
301 011220 104401 063174          11$:    TYPE    ,BLNKS8    ;TAB OVER THE PC COLUMN
302 011224 104401 063174          TYPE    ,BLNKS8    ;TAB OVER THE TEST NO. COLUMN
303 011230 104401 063174          TYPE    ,BLNKS8    ;TAB OVER THE RHAS UNIT COLUMN
304
305 011234 022737 177777 001126 12$:    CMP      #-1,$BDDAT ;DOES 'NED' UNIT TABLE LOCATION = -1 ?
306 011242 001404          BEQ      9$        ;YES...DON'T TYPE IT
307 011244 013746 001126          MOV      $BDDAT,-(SP) ;SAVE $BDDAT FOR TYPEOUT
      011250 104403          TYPOS    ;GO TYPE--OCTAL ASCII
      011252      006          .BYTE    6          ;TYPE 6 DIGITS
      011253      000          .BYTE    0          ;SUPPRESS LEADING ZEROS
308
309 011254 104401 001223          9$:     TYPE    ,$CRLF    ;FOR THE NEXT LINE IN BOTH TABLES
310 011260 005303          DEC      R3        ;COUNT DOWN 2 TABLES LOCATION COUNTER
311 011262 001327          BNE      14$      ;IF NOT = 0 TYPE OUT NEXT 2 LOCATIONS
312 011264 062706 000014          ADD      #14,SP    ;ADJUST STACK FOR NO 'POP' & RTS FROM 'COMPAR'
313
314 011270          13$:
315
316          ;:*****
317          ;:IN THE ABOVE TEST BITS 0,1,2, AND BIT 12 ARE TESTED
318
319          ;:IF THE 'DRIVES PRESENT' TYPE OUT DOES NOT AGREE WITH WHAT WAS
320          ;:FOUND USING RHER1 & RHAS, THEN THE ERROR IS IN THE LOGIC
321          ;:FOR BIT12(NED), OR UNIT SELECT(BIT 0 TO 2), OR RHER1, OR RHAS
322          ;:
323          ;:IT IS NOT POSSIBLE BY PROGRAM TO CHECK IF A NON-EXISTENT
324          ;:DRIVE IS REALLY STANDING THERE OR NOT
325          ;:
326          ;:MANUALLY LOAD LOCATION 'ERUNIT' WITH A UNIT NUMBER
327          ;:AND RESTART AT LOCATION 'ERSTAR' THIS WILL LOOP FOR
328          ;:EVER DOING EXACTLY AS TEST ON THAT ONE UNIT
329
330          ;:TO GET BACK TO MAIN DIAGNOSTIC HIT HALT SWITCH AND
    
```

331
332
333
344

```
:RESTART PROGRAM IN NORMAL MANNER
:*****
:*****
:*TEST 24 CONTROL AND STATUS 2 (RHCS2) - 'CLR'
:*THIS TESTS THE UNIT SELECT BITS (US1-4) AND CLEAR BIT #5 (CLR)

:*ALL REGISTERS ARE LOADED WITH ALL ONES EXCEPT BIT #0 AND #6
:*WHICH ARE 'GO' AND 'INTERRUPT ENABLE', THEN 'CLR' IS GIVEN
:*(RHDB IS READ FIRST AS THIS WILL SET DTL IN RHCS2 AND
:*SC AND TRE IN RHCS1.)

:*ANOTHER CLR IS GIVEN THEN ALL OTHER REGISTERS ARE READ
:*****
```

345 011270 000004
346 011272 004737 041460
347 011276 005037 001406

```
TST24: SCOPE
JSR PC,CLDISK ;SET REGISTERS AND CLEAR
CLR [RFLG$ ;CLEAR ANY ERRORS
```

348
349

```
:FILL ALL POSSIBLE BITS WITH ONES
```

350 011302 012777 177777 167720
351 011310 012777 177777 167714
352 011316 012777 177777 167710
353 011324 052777 157010 167704
354 011332 012777 001476 167700
355 011340 012777 177777 167674
356 011346 012777 017437 167670
357 011354 012777 177777 167664
358 011362 012777 016277 167660
359 011370 012777 177777 167654
360 011376 012777 177777 167650
361 011404 012777 000001 167646
362 011412 012777 177777 167640

```
MOV #177777,@RHDB ;BUS ADDRESS REGISTER GETS 177777
MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777
BIS #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 157010
MOV #1476,@RHCS1 ;CONTROL AND STATUS REGISTER/GETS 1476
MOV #177777,@RHER1 ;ERROR REGISTER1 GETS 177777
MOV #17437,@RHDST ;DESIRED SECTOR TRACK
MOV #177777,@RHER2 ;ERROR REGISTER 2
MOV #16277,@RHOF ;OFFSET REGISTER
MOV #177777,@RHCA ;DESIRED CYLINDER
MOV #177777,@RHER3 ;ERROR REGISTER 3
MOV #DMD,@RHMR ;MAINTENANCE REGISTER
MOV #177777,@RHMR ;MAINTENANCE REGISTER
```

363
364 011420 052712 000040
365 011424 013712 001374
366 011430 012700 001230

```
BIS #CLR,@R2 ;CLEAR ALL POSSIBLE BITS
MOV UNIT,@R2 ;REINSTATE UNIT NO.
MOV #RHDB,R0 ;R0 CONTAINS ADDR. OF ADDR. OF REG.
```

367
368
369

```
:DATA BUFFER REGISTER
```

370 011434 012737 177777 001124
371 011442 011037 041130
372 011446 013037 001126
373 011452 023737 001124 001126
374 011460 001401
375 011462 104001

```
MOV #177777,$GDDAT ;GOOD DATA FOR ERROR
MOV @R0,REGADR ;REGISTER ADDRESS
MOV @R0+,$BDDAT ;TEST DATA
CMP $GDDAT,$BDDAT ;COMPARE GOOD WITH TEST DATA
BEQ 2$ ;BRANCH IF GOOD
EMT 1
```

376
377

```
:AFTER A CLR IN RHCS2
```

378 011464 052712 000040
379
380
381 011470 013712 001374

```
2$: BIS #CLR,@R2 ;SET CLEAR AGAIN BECAUSE
;READING RHDB AFTER CLEARING WILL
;SET DLT SC AND TRE
MOV UNIT,@R2 ;REINSTATE UNIT NO.
```

382
383
384
385

```
:WORD COUNT REGISTER
```

011474 012737 177777 001124

```
3$: MOV #177777,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
```



```

011502 013037 001126      MOV    @(R0)+,$BDDAT  ;TEST DATA
011506 022737 177777 001126  CMP    #177777,$BDDAT ;COMPARE DATA
011514 001402          BEQ    4$             ;BRANCH IF GOOD
011516 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

386
387
388
389
;BUS ADDRESS REGISTER

```

011522 012737 000000 001124 4$:  MOV    #0,$GDDAT      ;GOOD DATA FOR ERROR TYPEOUT
011530 013037 001126          MOV    @(R0)+,$BDDAT  ;TEST DATA
011534 022737 000000 001126  CMP    #0,$BDDAT     ;COMPARE DATA
011542 001402          BEQ    5$             ;BRANCH IF GOOD
011544 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

390
391
392
393
;CONTROL AND STATUS 2 REGISTER

```

011550 012746 000100          5$:  MOV    #100,-(SP)     ;INCLUDE IR
011554 053716 001374          BIS    UNIT,(SP)     ;SET UNIT NO.
011560 012637 001124          MOV    (SP)+,$GDDAT  ;GOOD DATA FOR TYPE OUT
011564 013037 001126          MOV    @(R0)+,$BDDAT  ;TEST DATA
011570 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE DATA
011576 001402          BEQ    6$             ;BRANCH IF GOOD
011600 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

394
395
396
397
;CONTROL AND STATUS 1 REGISTER

```

011604 012737 004276 001124 6$:  MOV    #4276,$GDDAT  ;GOOD DATA FOR ERROR TYPEOUT
011612 013037 001126          MOV    @(R0)+,$BDDAT  ;TEST DATA
011616 022737 004276 001126  CMP    #4276,$BDDAT  ;COMPARE DATA
011624 001402          BEQ    7$             ;BRANCH IF GOOD
011626 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

398
399
400
401
;ERROR 1 REGISTER

```

011632 012737 000000 001124 7$:  MOV    #0,$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
011640 013037 001126          MOV    @(R0)+,$BDDAT  ;TEST DATA
011644 022737 000000 001126  CMP    #0,$BDDAT     ;COMPARE DATA
011652 001402          BEQ    10$            ;BRANCH IF GOOD
011654 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

402
403
404
405
;DESIRED SECTOR/TRACK REGISTER

```

011660 012737 017437 001124 10$: MOV    #17437,$GDDAT  ;GOOD DATA FOR ERROR TYPEOUT
011666 013037 001126          MOV    @(R0)+,$BDDAT  ;TEST DATA
011672 022737 017437 001126  CMP    #17437,$BDDAT ;COMPARE DATA
011700 001402          BEQ    11$            ;BRANCH IF GOOD
011702 004737 012302      JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
  
```

406
407
408
409

:ERROR 2 REGISTER

```

011706 012737 000000 001124 11$:  MOV    #0,$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
011714 013037 001126             MOV    @(R0)+,$BDDAT  ;TEST DATA
011720 022737 000000 001126     CMP    #0,$BDDAT     ;COMPARE DATA
011726 001402             BEQ    12$           ;BRANCH IF GOOD
011730 004737 012302             JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                           ;IN RHCS2
  
```

410
411
412
413

:OFFSET REGISTER

```

011734 012737 116000 001124 12$:  MOV    #116000,$GDDAT ;GOGD DATA FOR ERROR TYPEOUT
011742 013037 001126             MOV    @(R0)+,$BDDAT  ;TEST DATA
011746 022737 116000 001126     CMP    #116000,$BDDAT ;COMPARE DATA
011754 001402             BEQ    13$           ;BRANCH IF GOOD
011756 004737 012302             JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                           ;IN RHCS2
  
```

414
415
416
417

:DESIRED CYLINDER ADDRESS REGISTER

```

011762 012737 001777 001124 13$:  MOV    #1777,$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
011770 013037 001126             MOV    @(R0)+,$BDDAT  ;TEST DATA
011774 022737 001777 001126     CMP    #1777,$BDDAT   ;COMPARE DATA
012002 001402             BEQ    14$           ;BRANCH IF GOOD
012004 004737 012302             JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                           ;IN RHCS2
  
```

418
419
420
421

:ERROR 3 REGISTER

```

012010 012737 000000 001124 14$:  MOV    #0,$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
012016 013037 001126             MOV    @(R0)+,$BDDAT  ;TEST DATA
012022 022737 000000 001126     CMP    #0,$BDDAT     ;COMPARE DATA
012030 001402             BEQ    15$           ;BRANCH IF GOOD
012032 004737 012302             JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                           ;IN RHCS2
  
```

422
423
424

:ATTENTION SUMMARY REGISTER

```

425 012036 013037 001126             15$:  MOV    @(R0)+,$BDDAT  ;GET RHAS CONTENTS
426 012042 012737 000000 001124     MOV    #0,$GDDAT     ;GOOD DATA FOR ERROR TYPE OUT
427 012050 123737 001124 001126     CMPB   $GDDAT,$BDDAT ;COMPARE FOR RHAS
428 012056 001402             BEQ    16$           ;BRANCH IF GOOD
429 012060 004737 012302             JSR    PC,ERCS2C     ;JUMP TO ERROR FOR CLR (BIT 5)
                                           ;IN RHCS2
  
```

430
431
432
433
434

:MAINTAINABILITY REGISTER

```

012064 012737 000400 001124 16$:  MOV    #400,$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
012072 013037 001126             MOV    @(R0)+,$BDDAT  ;TEST DATA
012076 022737 000400 001126     CMP    #400,$BDDAT   ;COMPARE DATA
012104 001402             BEQ    17$           ;BRANCH IF GOOD
  
```

```
012106 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
;IN RHCS2
435
436 ;DRIVE STATUS REGISTER
437
438 012112 012737 000600 001124 17$: MOV #600,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
439 012120 013046 MOV @ (R0)+, -(SP) ;GET RHDS1
440 012122 011637 001126 MOV (SP), $BDDAT ;TEST DATA
441 012126 042716 001100 BIC #VV!PROG, (SP) ;CLEAR VV AND PROG
442 012132 022726 000600 CMP #600, (SP)+ ;COMPARE DATA
443 012136 001402 BEQ 20$ ;BRANCH IF GOOD
444 012140 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
445 ;IN RHCS2
446
447 ;DRIVE TYPE
448
449 012144 013737 001410 001124 20$: MOV SAVDT,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
012152 013037 001126 MOV @ (R0)+, $BDDAT ;TEST DATA
012156 023737 001410 001126 CMP SAVDT,$BDDAT ;COMPARE DATA
012164 001402 BEQ 21$ ;BRANCH IF GOOD
012166 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
;IN RHCS2
450
451 ;SERIAL NUMBER REGISTER
452
453
454 012172 013737 001412 001124 21$: MOV SAVSN,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
012200 013037 001126 MOV @ (R0)+, $BDDAT ;TEST DATA
012204 023737 001412 001126 CMP SAVSN,$BDDAT ;COMPARE DATA
012212 001402 BEQ 22$ ;BRANCH IF GOOD
012214 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
;IN RHCS2
455
456 ;FCC1 POSITION
457
458 012220 012737 000000 001124 22$: MOV #0,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
012226 013037 001126 MOV @ (R0)+, $BDDAT ;TEST DATA
012232 022737 000000 001126 CMP #0,$BDDAT ;COMPARE DATA
012240 001402 BEQ 23$ ;BRANCH IF GOOD
012242 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
;IN RHCS2
459
460 ;ECC2 PATTERN
461
462
463 012246 012737 000000 001124 23$: MOV #0,$GDDAT ;GOOD DATA FOR ERROR TYPEOUT
012254 013037 001126 MOV @ (R0)+, $BDDAT ;TEST DATA
012260 022737 000000 001126 CMP #0,$BDDAT ;COMPARE DATA
012266 001402 BEQ 24$ ;BRANCH IF GOOD
012270 004737 012302 JSR PC,ERCS2C ;JUMP TO ERROR FOR CLR (BIT 5)
;IN RHCS2
464
465
```

```

466          ;LOOK-AHEAD REGISTER
467
468 012274 005720 24$: TST (R0)+ ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
469                                     ;AFTER AN INIT IT IS NOT CHECKED
470
471          ;CURRENT CYLINDER ADDRESS REGISTER
472
473 012276 005720 25$: TST (R0)+ ;AS THE CURRENT REG. CANNOT BE PREDICTED
474                                     ;AFTER A INIT IT IS NOT CHECKED
475
476 012300 000405 26$: BR TST25 ;BRANCH OVER JSR
477
478
479 012302 014037 041130 ERCS2C: MOV -(R0),REGADR ;FAILING REGISTER ADDRESS
480 012306 104001 EMT 1
481                                     ;NOT CLEAR APPROPRIATE BITS
482                                     ;OR CLEARED EXTRA BITS
483 012310 005720 TST (R0)+ ;UNDO -(R0) FOR BAD DATA
484 012312 000207 RTS PC ;RETURN TO TEST ABOVE
485
492          ;:*****
          ;*TEST 25 PACK ACKNOWLEDGE COMMAND TEST
          ;*THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
          ;*THEN ALL REGISTERS WILL BE CHECKED
          ;*RH CLEAR WILL BE GIVEN
          ;*THEN ALL REGISTERS WILL BE CHECKED
          ;:*****
493 012314 000004 TST25: SCOPE
494 012316 012706 001100 MOV #STACK,SP ;RESET STACK
495 012322 012737 000025 001432 MOV #25,TSTNM ;MOVE #25 TO TEST NUMBER
496 012330 004737 041460 JSR PC,CLDISK ;INIT AND SET UP GENERAL CPU/DEVICE
497                                     ;REGISTER CORRESPONDENCE AND UNIT NO.
498 012334 012777 000001 166716 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
499 012342 013777 001500 166670 MOV PKACK,@RHCS1 ;LOAD 'PACK ACKNOWLEDGE COMMAND' INTO RHCS1
500
501          ;SAVE REGISTERS FOR COMPARISON AFTER 'GO' IS ISSUED
502 012350 004037 042152 JSR R0,SAVER ;SAVE
503 012354 001232 RHWC ;FROM
504 012356 002554 REINTO ;TO
505 012360 000023 19. ;NUMBER OF REGISTERS SAVED
506
507
508 012362 052777 000001 166650 BIS #GO,@RHCS1 ;ISSUE 'GO' TO PACK ACKNOWLEDGE COMMAND
509
510          ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
511 012370 052737 000100 002604 BIS #VV,REINTO+30 ;SAVED RHDS1
512
513          ;AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
514          ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
515          ;BE DONE
516
517 012376 004037 042152 JSR R0,SAVER ;SAVE
518 012402 001232 RHWC ;FROM
519 012404 001510 WRFROM
520 012406 000023 19. ;NUMBER OF REGISTERS SAVED
    
```

```

521
522           ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
523           ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
524           ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
525 012410 113737 002601 001535      MOVB    REINTO+25,WRFROM+25;SAVE UPPER RHAS
526
527
528           ;COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
529           ;WITH AFTER GO
530
531 012416 004037 042354              JSR     R0,COMPARE      ;COMPARE
532 012422 002554                    REINTO                    ;GOOD BUFFER
533 012424 001510                    WRFROM                    ;TEST BUFFER
534 012426 000023                    19.                      ;NUMBER
535 012430 012436                    1$                       ;RETURN FOR ERROR
536 012432 012436                    1$                       ;SAME
537 012434 012456                    2$                       ;RETURN FOR GOOD COMPARISON
538
539 012436 013705 046426      1$:    MOV     ERWORD,R5      ;GETTING READY TO INDEX
540 012442 060505                    ADD     R5,R5           ;DOUBLE ERROR WORD
541 012444 016537 001230 041130    MOV     RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
542
543 012452 104001                    EMT     1
544
545
546 012454 000207                    RTS     PC              ;AFTER PACK ACKNOWLEDGE COMMAND
547
548 012456      2$:                  ;WITH GO IS GIVEN
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563 012456 000004                    TST26: SCOPE
564 012460 012706 001100                    MOV     #STACK,SP      ;RESET STACK
565 012464 012737 000026 001432    MOV     #26,TSTNM     ;MOVE #26 TO TEST NUMBER
566 012472 004737 041460                    JSR     PC,CLDISK     ;INIT AND SET UP GENERAL CPU/DEVICE
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
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
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
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

;BEFORE RESET SAVE REGISTERS IN READ INTO BUFFER

```

577 012606 004037 042152      JSR      R0,SAVER      ;SAVE
578 012612 001232              RHWC              ;FROM
579 012614 002554              REINTO          ;TO
580 012616 000021              17.              ;NUMBER
581
582      ;GIVE RESET AND REINSTATE UNIT NUMBER
583 012620 000005              RESET
584 012622 004737 054216      JSR      PC,$TKINT    ;INITIALIZE TK
585 012626 053777 001374 166402  BIS      UNIT,@RHCS2
586
587      ;CHANGE ORIGINAL SAVED REGISTERS TO EXPECTED VALUES AFTER RESET
588 012634 005037 002556      CLR      REINTO+2     ;CLEAR SAVED RHBA
589 012640 013746 001374      MOV      UNIT,-(SP)   ;GET UNIT NUMBER FRO SAVED RHCS2
590 012644 052716 000100      BIS      #IR,(SP)    ;INCLUDE IR
591 012650 012637 002560      MOV      (SP)+,REINTO+4 ;SAVED RHCS2
592 012654 012737 004276 002562  MOV      #DVA!RDY!76,REINTO+6 ;SAVED RHCS1
593 012662 005037 002564      CLR      REINTO+10    ;SAVED RHER1
594 012666 005037 002570      CLR      REINTO+14    ;SAVED RHER2
595 012672 012737 116000 002572  MOV      #116000,REINTO+16 ;SAVED RHOF
596 012700 005037 002576      CLR      REINTO+22    ;SAVED RHER3
597 012704 105037 002600      CLR      REINTO+24    ;SAVED RHAS
598 012710 012737 000400 002602  MOV      #400,REINTO+26;SAVED RHMR
599
600      ;CHANGE RHDS1 WITHOUT CHANGING PROG BIT
601 012716 013746 002604      MOV      REINTO+30,-(SP) ;GET RHDS1
602 012722 042716 176777      BIC      #^CPROG,(SP)  ;CLEAR EVERYTHING EXCEPT PROG
603 012726 052716 000700      BIS      #700,(SP)    ;SET EXPECTED BITS - 'DPR', 'DRY' & 'VV'
604
605 012732 012637 002604      4$:      MOV      (SP)+,REINTO+30;SAVED RHDS1
606 012736 005037 002612      CLR      REINTO+36    ;SAVED RHEC1
607 012742 005037 002614      CLR      REINTO+40    ;SAVED RHEC2
608
609      ;AFTER RESET, SAVE REGISTERS FOR COMPARISONS TO BE DONE
610 012746 004037 042152      JSR      R0,SAVER      ;SAVE
611 012752 001232              RHWC              ;FROM
612 012754 001510              WRFROM          ;TO
613 012756 000021              17.              ;NUMBER
614
615      ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
616      ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
617      ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
618 012760 113737 002601 001535  MOV      REINTO+25,WRFROM+25;SAVE UPPER RHAS
619
620      ;COMPARE REGISTERS BEFORE RESET WITH REGISTERS AFTER RESET
621 012766 004037 042354      JSR      R0,COMPAR     ;COMPARE
622 012772 002554              REINTO          ;GOOD BUFFER
623 012774 001510              WRFROM          ;TEST BUFFER
624 012776 000021              17.              ;NUMBER
625 013000 013006              1$              ;RETURN FOR ERROR
626 013002 013006              1$              ;SAME
627 013004 013026              2$              ;RETURN FOR GOOD COMPARISON
628
629 013006 013705 046426      1$:      MOV      ERWORD,R5    ;GETTING READY TO INDEX
630 013012 060505              ADD      R5,R5      ;DOUBLE ERROR WORD
631 013014 016537 001230 041130  MOV      RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
632 013022 104001              EMT      1
633
;A RESET THAT IS AN

```

634
635
636 013024 000207
637 013026

2\$: RTS PC

:UNIBUS INITIALIZE CAUSED
:AN IMPROPER REGISTER CHANGE
:RETURN TO COMPARISON
:RETURN TO POINT ON GOOD COMPARISON


```

1      .SBTTL  SILO TESTS
2
10     ;*****
;TEST 27      SILO TST 1
;THIS TESTS THE SILO BUFFER IN THE RH11 CONTROLLER
;A READ IS ATTEMPTED FROM AN EMPTY SILO
;DATA LATE (DLT) (RHCS2), TRANSFER ERROR (TRE) (RHCS1),
;SPECIAL CONDITION (SC) (RHCS1) SHOULD SET
;THEN LOADING '1' INTO TRE SHOULD CLEAR DLT, TRE AND SC
;*****
11     013026 000004
12     013030 012706 001100
13     013034 012737 000027 001432
;TST27: SCOPE
;MOV #STACK,SP ;RESET STACK
;MOV #27,TSTNM ;MOVE #27 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

14     013042 005737 001436
15     013046 001402
16     013050 000137 013204
17     013054
18     013054 004737 041460
19     013060 017700 166144
20     013064 013746 001374
21     013070 052716 100100
22     013074 004737 041064
23     013100 022637 001312
24     013104 001403
25     013106 010237 001122
26     013112 104011
27
28     013114 022737 144200 001314 1$:
29
30     013122 001403
31     013124 010137 001122
32     013130 104011
33
34
35
36
37     013132 012711 040000 2$:
38
39     013136 004737 041064
40     013142 022737 004200 001314
41
42     013150 001403
43     013152 010137 001122
44     013156 104011
45
46
47     013160 013746 001374
48     013164 052716 000100
49     013170 022637 001312
50     013174 001403

```

```

51 013176 010237 001122      MOV      R2,$BDADR      ;STORE FAILING ADDR
52 013202 104011              EMT      11
53
65
:*****
:*TEST 30      SILO TEST 2
:*THIS TESTS THE IR AND 'OR' BITS OF RHCS2
:*AT THE BEGINNING IR SHOULD BE SET AND 'OR' RESET
:*LOADING 0 IN SILO RESETS IR FOR ONLY 2 MICRO SECONDS
:*THIS TIME CANNOT BE CHECKED BUT IT IS CHECKED TO SEE IF
:*IT DOES GO DOWN OR NOT
:*THEN ALL 1 IS LOADED IN SILO 'OR' SHOULD BECOME SET
:*IN 30 MICRO SECONDS AGAIN TIME IS NOT CHECKED
:*'OR' SHOULD BE SET
:*THE OUTPUT FROM THE SILO SHOULD BE 0 AND ALL ONES
:*****
TST30:  SCOPE
66 013204 000004              MOV      #STACK,SP      ;RESET STACK
67 013206 012706 001100      MOV      #30,TSTNM      ;MOVE #30 TO TEST NUMBER
68 013212 012737 000030 001432
;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
      TST      RH70      ;TEST FLAG FOR RH70 CONTROLLER
      BEQ     64$      ;IF FLAG = 1, THIS TEST IS SKIPPED
      JMP     TST31     ;JUMP TO NEXT TEST
69 013232 004737 041460      JSR     PC,CLDISK      ;IF FLAG = 0, DO THIS TEST
70                                     ;CLEAR REGISTERS LOAD R'S
71 013236 013746 001374      MOV     UNIT,-(SP)
72 013242 052716 000100      BIS     #IR,(SP)
73 013246 004737 041064      JSR     PC,PUTREG      ;SAVE REGISTERS
74 013252 022637 001312      CMP     (SP)+,CS2      ;IR SHOULD BE SET 'OR' RESET
75 013256 001403              BEQ     1$
76 013260 010237 001122      MOV     R2,$BDADR      ;FAILING REGISTER RHCS2
77 013264 104011              EMT     11
78                                     ;SET, UNIT NO. SET AND
79                                     ;ALL OTHER BITS 0
80 013266 005077 165736      1$:    CLR     @RHDB      ;LOAD DATA BUFFER (SILO) WITH 0
81 013272 012777 177777 165730  MOV     #-1,@RHDB      ;LOAD SILO WITH ALL ONES
82 013300 013737 001236 013310  MOV     RHCS2,2$      ;ADDRESS OF RHCS2
83 013306 104415              WAI     ;WAIT TRAP
84 013310 000000      2$:    .WORD ;ADDRESS OF RHCS2
85 013312 000200              OR
86 013314 013746 001374      3$:    MOV     UNIT,-(SP)
87 013320 052716 000300      BIS     #OR!IR,(SP)   ;IR AND 'OR'
88 013324 004737 041064      JSR     PC,PUTREG      ;SAVE REGISTERS
89 013330 022637 001312      CMP     (SP)+,CS2      ;IR AND 'OR' SHOULD BE SET
90 013334 001403              BEQ     4$
91 013336 010237 001122      MOV     R2,$BDADR      ;SAVE RHCS2 ADDR. FAILING REG.
92 013342 104011              EMT     11
93                                     ;SET TOGETHER WITH IR AND
94                                     ;UNIT NO.
95 013344 017700 165660      4$:    MOV     @RHDB,R0      ;SAVE SILO DATA SHOULD BE 0
96 013350 017705 165654      MOV     @RHDB,R5      ;SAVE SILO DATA SHOULD BE ALL 1
97 013354 022700 000000      CMP     #0,R0          ;FIRST WORD 0? XYZ DO MORE TEST
98 013360 001410              BEQ     5$             ;BRANCH IF YES
99 013362 005037 001124      CLR     $GDDAT ;GOOD DATA
100

```

```

101 013366 010037 001126      MOV    R0,$BDDAT      ;BAD DATA
102 013372 013737 001230 041130  MOV    RHDB,REGADR   ;SAVE RHDB FAILING REG.
103 013400 104001                EMT    1
104                                ;'0' WHEN 'OR' WAS SET
105 013402 022705 177777      5$:   CMP    #-1,R5      ;SECOND WORD ALL ONES?
106 013406 001411                BEQ    TST31         ;BRANCH IF YES
107 013410 012737 177777 001124  MOV    #-1,$GDDAT   ;GOOD DATA
108 013416 010537 001126      MOV    R5,$BDDAT    ;BAD DATA
109 013422 013737 001230 041130  MOV    RHDB,REGADR   ;SAVE RHDB FAILING REG.
110 013430 104001                EMT    1
111                                ;WORD OF ALL ONES WHEN 'OR'
112                                ;WAS SET
113
119
:*****
:*TEST 31      SILO TEST 3
:*THIS TESTS SILO BUFFER BY FILLING IT WITH A COUNT FROM
:*0 TO 65 AND THEN CHECKING IF IR IS DOWN AND 'OR'
:*IS HIGH AND COMPARING THE SILO OUTPUT.
:*****
120 013432 000004                TST31:  SCOPE
121 013434 012737 000031 001432  MOV    #31,TSTNM     ;MOVE #31 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

013442 005737 001436      TST    RH70          ;TEST FLAG FOR RH70 CONTROLLER
013446 001402                BEQ    64$           ;IF FLAG = 1, THIS TEST IS SKIPPED
013450 000137 013672      64$:   JMP    TST32         ;JUMP TO NEXT TEST
013454                                ;IF FLAG = 0, DO THIS TEST

122
123 013454 012700 050224      MOV    #SILOTB,R0    ;TABLE POINTER
124 013460 012705 000103      MOV    #66.,R5      ;COUNTER
125 013464 005020      1$:   CLR    (R0)+        ;CLEAR TOTAL TABLE
126 013466 005305                DEC    R5            ;COUNT
127 013470 001375                BNE    1$           ;BRANCH IF NOT COMPLETELY CLEAR
128 013472 004737 041460      JSR    PC,CLDISK    ;CLEAR ALL REG.
129 013476 005000                CLR    R0
130 013500 012705 000102      MOV    #66.,R5      ;COUNT
131 013504 010077 165520      2$:   MOV    R0,@RHDB    ;LOAD SILO WITH COUNT FROM 0 TO 65
132 013510 005200                INC    R0            ;NEXT COUNT
133 013512 005305                DEC    R5            ;IS 66 LOADS DONE?
134 013514 001373                BNE    2$           ;BRANCH IF NOT.
135 013516 013746 001374      MOV    UNIT,-(SP)
136 013522 052716 000200      BIS    #OR,(SP)
137 013526 004737 041064      JSR    PC,PUTREG    ;SAVE REGISTERS
138 013532 022637 001312      CMP    (SP)+,CS2    ;'OR' SHOULD BE SET IR RESET
139 013536 001405                BEQ    3$           ;BRANCH IF YES
140 013540 010237 001122      MOV    R2,$BDADR    ;SAVE RHCS2 ADR. FAILING REG.
141 013544 104011                EMT    11
142 013546 005037 001406      CLR    ERFLG$      ;RESET AFTER SILO WAS FULL
143 013552 012700 050224      3$:   MOV    #SILOTB,R0  ;POINTER
144 013556 012705 000102      MOV    #66.,R5      ;COUNTER
145 013562 017720 165442      4$:   MOV    @RHDB,(R0)+ ;READ SILO
146 013566 005305                DEC    R5            ;COUNT
147 013570 001374                BNE    4$           ;BRANCH IF 66 NOT DONE
148 013572 012700 050224      MOV    #SILOTB,R0  ;POINTER
149 013576 012705 000102      MOV    #66.,R5
150 013602 005046      CLR    -(SP)
    
```

```

151 013604 021620 5$: CMP (SP),(R0)+
152 013606 001425 BEQ 8$ ;BRANCH IF GOOD
153 013610 014037 001126 MOV -(R0), $BDDAT ;BAD DATA
154 013614 011637 001124 MOV (SP), $GDDAT ;GOOD DATA
155 013620 013737 001230 041130 MOV RHDB, REGADR ;FAILING REG. RHDB
156 013626 005737 001406 TST ERFLG$ ; IS THIS FIRST ERROR?
157 013632 001002 BNE 6$ ;IF NOT BRANCH
158 013634 104012 EMT 12
159 013636 000401 BR 7$ ;BRANCH TO AVOID PRINTING NEXT ERROR
160 013640 6$: EMT 13
161 ;HAD A COUNT WRITTEN IN.
162 ;ON READ OUT AN ERROR WAS
163 ;DETECTED. THE TOTAL SILO
164 ;READOUT IS IN LOCATION
165 ;"SILOTB" TO THE NEXT 65
166 ;WORDS.
167 013642 005720 7$: TST (R0)+ ;INCREMENT (R0)
168 ;ARE FURTHER COMPARES TO
169 013644 017746 165270 MOV @SWR, -(SP) ;BE DONE
170 013650 042716 177577 BIC #^CSW07!SW08, (SP) ;ONLY KEEP SW7 AND SW8
171 013654 022726 000200 CMP #SW07, (SP)+ ;TEST SW07
172 013660 001403 BEQ 9$ ;IF NO MORE COMPARE THEN BRANCH
173 013662 005216 8$: INC (SP) ;NEXT GOOD WORD
174 013664 005305 DEC R5 ;COUNT
175 013666 001346 BNE 5$ ;BRANCH IF 66 NOT COMPLETE
176 013670 005726 9$: TST (SP)+ ;POP STACK
177
182 ;*****
; *TEST 32 SILO TEST4
; *NOW PUT 67 WORDS INTO SILO AND CHECK FOR DLT
; *EVEN AFTER THE 67TH. WORD INPUT THE FIRST WORD SHOULD NOT CHANGE
;*****
183 013672 000004 TST32: SCOPE
184 013674 012737 000032 001432 MOV #32, TSTNM ;MOVE #32 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

013702 005737 001436 TST RH70 ;TEST FLAG FOR RH70 CONTROLLER
013706 001402 BEQ 64$ ;IF FLAG = 1, THIS TEST IS SKIPPED
013710 000137 014016 JMP TST33 ;JUMP TO NEXT TEST
013714 64$: JSR PC, CLDISK ;IF FLAG = 0, DO THIS TEST
;CLEAR DISK REG.

185 013714 004737 041460 JSR PC, CLDISK
186
187 013720 005000 CLR R0 ;CLEAR R0
188 013722 005200 1$: INC R0 ;ADD 1
189 013724 010077 165300 MOV R0, @RHDB ;LOAD SILO
190 013730 022700 000103 CMP #67, R0 ;67 DONE?
191 013734 001401 BEQ 2$ ;BRANCH IF YES
192 013736 000771 BR 1$ ;NO SO BRANCH
193 013740 004737 041064 2$: JSR PC, PUTREG ;SAVE REGISTERS
194
195 013744 032737 100000 001312 BIT #DLT, CS2 ;DLT SET?
196 013752 001003 BNE 3$ ;BRANCH IF YES
197 013754 010237 001122 MOV R2, $BDADR ;FAILING ADDRESS RHCS2
198 013760 104011 EMT 11
199 013762 017737 165242 001126 3$: MOV @RHDB, $BDDAT ;INPUT TO SILO
    
```

```
200 013770 012737 000001 001124      MOV    #1,$GDDAT      ;GOOD DATA
201 013776 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE
202 014004 001404          BEQ    TST33          ;BRANCH IF GOOD
203 014006 013737 001230 041130      MOV    RHDB,REGADR   ;FAILING REG. RHDB
204 014014 104012          EMT    12
205                                     ;AFTER THE 67TH INPUT.
206
213                                     ;*****
;TEST 33      SILO TEST 5
;THE SILO IS LOADED WITH 0,1,2,3 THEN AFTER
;'OR' IS UP A CLR IN RHCS2 IS DONE THEN 4,
;IS LOADED. AFTER 'OR' IS UP 2 READS FROM
;SILO ARE DONE, ON THE LAST, 'DTL' IN RHCS2 SHOULD BE SET.
;*****
214 014016 000004          TST33: SCOPE
215 014020 012737 000033 001432      MOV    #33,TSTNM     ;MOVE #33 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

014026 005737 001436      TST    RH70          ;TEST FLAG FOR RH70 CONTROLLER
014032 001402          BEQ    64$           ;IF FLAG = 1, THIS TEST IS SKIPPED
014034 000137 014316      JMP    TST34         ;JUMP TO NEXT TEST
014040          64$:
216 014040 004737 041460      JSR    PC,CLDISK    ;CLEAR DISK
217
218 014044 013746 001374      MOV    UNIT,-(SP)   ;GET UNIT NO.
219 014050 052716 000100      BIS    #IR,(SP)     ;SET INPUT READY
220 014054 004737 041064      JSR    PC,PUTREG    ;SAVE REGISTERS
221 014060 022637 001312      CMP    (SP)+,CS2    ;IR SHOULD BE SET 'DR' CLEARED
222 014064 001403          BEQ    1$           ;BRANCH IF GOOD
223 014066 010237 001122      MOV    R2,$BDADR   ;FAILING REGISTER RHCS2
224 014072 104011          EMT    11
225                                     ;AND ALL OTHER BITS 0
226 014074 013700 001230      1$: MOV    RHDB,R0 ;R0 HAS RHDB ADDRESS
227 014100 005001          CLR    R1           ;DATA
228 014102 010110          2$: MOV    R1,@R0    ;0, THEN 1 THEN 2 THEN 3
229                                     ;IN RHDB
230 014104 005201          INC    R1           ;INCREMENT DATA
231 014106 022701 000004      CMP    #4,R1        ;IS 4 DONE
232 014112 103373          BHIS  2$           ;BRANCH IF NOT
233 014114 013737 001236 014124      MOV    RHCS2,3$
234 014122 104415          WAT
235 014124 000000          3$: .WORD 0
236 014126 000200          OR
237 014130 004737 041460      JSR    PC,CLDISK    ;CLR IN RHCS2
238 014134 013746 001374      MOV    UNIT,-(SP)   ;UNIT NO.
239 014140 052716 000100      BIS    #IR,(SP)
240 014144 004737 041064      JSR    PC,PUTREG    ;SAVE REGISTERS
241 014150 022637 001312      CMP    (SP)+,CS2    ;IR SHOULD BE SET '0'=0
242 014154 001403          BEQ    4$           ;BRANCH IF GOOD
243 014156 010237 001122      MOV    R2,$BDADR   ;FAILING REGISTER RHCS2
244 014162 104011          EMT    11
245                                     ;AND ALL OTHER BITS 0
246 014164 013700 001230      4$: MOV    RHDB,R0 ;R0 HAS RHDB ADDRESS
247 014170 012710 000004      MOV    #4,@R0       ;LOAD 4 IN SILO
248 014174 011201          MOV    @R2,R1       ;SAVE RHCS2
249 014176 011005          MOV    @R0,R5       ;READ THE 4 IN SILO
```

```

250 014200 011003      MOV      @R0,R3      ;READ SILO TO GET DLT
251 014202 011204      MOV      @R2,R4      ;SAVE RHCS2
252 014204 032701 000200  BIT      #OR,R1      ;TEST FOR OR IN RHCS2
253 014210 001424      BEQ      6$          ;IF OR IS NOT SET BRANCH
254 014212 022705 000004  CMP      #4,R5      ;SILO 4 IS NOW COMPARED
255 014216 001410      BEQ      5$          ;
256 014220 010037 041130  MOV      R0,REGADR   ;SILO ADDRESS
257 014224 012737 000004 001124  MOV      #4,$GDDAT   ;GOOD DATA
258 014232 010537 001126  MOV      R5,$BDDAT   ;BAD DATA
259 014236 104001      EMT      1          ;
260                          ;PUT IN AFTER 'OR' WAS UP
261 014240 005703      5$:  TST      R3      ;IS IT ZERO BECAUSE SILO
262                          ;IS DESTRUCTIVE READ
263 014242 001407      BEQ      6$          ;BRANCH IF GOOD
264 014244 010037 041130  MOV      R0,REGADR   ;SILO ADDRESS
265 014250 005037 001124  CLR      $GDDAT      ;GOOD DATA
266 014254 010337 001126  MOV      R3,$BDDAT   ;BAD DATA
267 014260 104001      EMT      1          ;
268                          ;AFTER THE ONE WORD PUT IN
269                          ;HAS BEEN TAKEN OUT AS
270                          ;SILO IS A DESTRUCTIVE READ
271 014262 032704 100000  6$:  BIT      #DLT,R4  ;
272 014266 001013      BNE      TST34      ;BRANCH IF DLT SET
273 014270 013746 001374  MOV      UNIT,-(SP)  ;GET UNIT NO
274 014274 052716 100300  BIS      #DLT!OR!IR,(SP) ;
275 014300 012637 001124  MOV      (SP)+,$GDDAT ;GOOD DATA
276 014304 010437 001126  MOV      R4,$BDDAT   ;BAD DATA
277 014310 010237 041130  MOV      R2,REGADR   ;RHCS2 ADDRESS
278 014314 104001      EMT      1          ;
    
```

```
1      .SBTTL  MORE REGISTER TESTS
2
6      :*****
7      :*TEST 34  TEST ODD BYTE INSTRUCTION ON RHCS1 - RH11
8      :*RDY (BIT 07) AND DVA (BIT 11) SHOULD ALWAYS BE SET
9      :*****
10     TST34: SCOPE
11     014316 000004
12     014320 012737 000034 001432      MOV      #34,TSTNM      ;MOVE #34 TO TEST NUMBER
13
14     ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70
15
16     014326 005737 001436      TST      RH70          ;TEST FLAG FOR RH70 CONTROLLER
17     014332 001402              BEQ      64$           ;IF FLAG = 1, THIS TEST IS SKIPPED
18     014334 000137 014454      JMP      TST35        ;JUMP TO NEXT TEST
19     014340              64$:                ;IF FLAG = 0, DO THIS TEST
20     014340 012706 001100      MOV      #STACK,SP    ;RESET STACK
21     014344 004737 041460      JSR      PC,CLDISK    ;CLEAR DISK REG.
22
23     014350 012711 003566      MOV      #3566,@R1    ;LOAD RHCS1 WITH ANY NUMBER
24     014354 010146              MOV      R1,-(SP)     ;GETTING READY TO FORM ODD BYTE
25     014356 005216              INC      (SP)         ;SP NOW HAS ODD BYTE FOR RHCS1
26     014360 112736 000005      MOVSB   #5,@(SP)+     ;MOVE 5 INTO ODD BYTE FOR RHCS1
27     014364 011137 001126      MOV      @R1,$BDDAT   ;TEST DATA
28     014370 022737 006766 001126  CMP      #2566!DVA!RDY,$BDDAT ;RHCS1 SHOULD HAVE 6766
29     014376 001406              BEQ      1$           ;BRANCH IF GOOD
30     014400 012737 006766 001124  MOV      #2566!DVA!RDY,$GDDAT ;GOOD DATA
31     014406 010137 041130      MOV      R1,REGADR    ;FAILING REGISTER RHCS1
32     014412 104001              EMT      1
33
34
35     ;ODD BYTE OF RHCS1 GAVE
36     ;WRONG RESULTS
37
38     014414 112711 000032      1$:  MOVSB  #32,@R1      ;MOVE INTO EVEN BYTE
39     014420 011137 001126      MOV      @R1,$BDDAT   ;TEST DATA
40     014424 022737 006632 001126  CMP      #2432!DVA!RDY,$BDDAT ;RHCS1 SHOULD HAVE 6632
41     014432 001460              BEQ      TST36        ;DO NEXT RH11 TEST IF GOOD
42     014434 012737 006632 001124  MOV      #2432!DVA!RDY,$GDDAT ;GOOD DATA
43     014442 010137 041130      MOV      R1,REGADR    ;FAILING REGISTER RHCS1
44     014446 104001              EMT      1
45
46     ;BYTE OF RHCS1 GAVE WRONG
47     ;RESULT
48
49     014450 000137 014574      JMP      TST36        ;SKIP RH70 TEST
50
51     :*****
52     :*TEST 35  TEST ODD BYTE INSTRUCTION ON RHCS1 - RH70
53     :*RDY (BIT 07) AND DVA (BIT 11) SHOULD ALWAYS BE SET
54     :*****
55     TST35: SCOPE
56     014454 000004
57     014456 012737 000035 001432      MOV      #35,TSTNM    ;MOVE #35 TO TEST NUMBER
58     014464 012706 001100      MOV      #STACK,SP    ;RESET STACK
59     014470 004737 041460      JSR      PC,CLDISK    ;CLEAR DISK REG.
60
61     014474 012711 003566      MOV      #3566,@R1    ;LOAD RHCS1 WITH ANY NUMBER
62     014500 010146              MOV      R1,-(SP)     ;GETTING READY TO FORM ODD BYTE
63     014502 005216              INC      (SP)         ;SP NOW HAS ODD BYTE FOR RHCS1
64     014504 112736 000005      MOVSB   #5,@(SP)+     ;MOVE 5 INTO ODD BYTE FOR RHCS1
65     014510 011137 001126      MOV      @R1,$BDDAT   ;TEST DATA
```



```
50
51 014514 022737 004766 001126      CMP      #566!DVA!RDY,$BDDAT ;RHCS1 SHOULD HAVE 4766
52 014522 001406                    BEQ      1$                    ;BRANCH IF GOOD
53 014524 012737 004766 001124      MOV      #566!DVA!RDY,$GDDAT ;GOOD DATA
54 014532 010137 041130      MOV      R1,REGADR           ;FAILING REGISTER RHCS1
55 014536 104001                    EMT      1
56                                     ;ODD BYTE OF RHCS1 GAVE
57                                     ;WRONG RESULTS
58
59 014540 112711 000032      1$:     MOVB     #32,@R1           ;MOVE INTO EVEN BYTE
60 014544 011137 001126      MOV      @R1,$BDDAT          ;TEST DATA
61 014550 022737 004632 001126      CMP      #432!DVA!RDY,$BDDAT ;RHCS1 SHOULD HAVE 4632
62 014556 001406                    BEQ      TST36                ;BRANCH IF GOOD
63 014560 012737 004632 001124      MOV      #432!DVA!RDY,$GDDAT ;GOOD DATA
64 014566 010137 041130      MOV      R1,REGADR           ;FAILING REGISTER RHCS1
65 014572 104001                    EMT      1
66                                     ;BYTE OF RHCS1 GAVE WRONG
67                                     ;RESULTS
68
72                                     ;*****
72                                     ;*TEST 36      TEST ODD BYTE INSTRUCTION ON RHCS2
72                                     ;*IR (BIT 06) AND THE UNIT SELECT (BIT 0-2) WILL BE SET
72                                     ;*****
73 014574 000004                    TST36: SCOPE
74 014576 012737 000036 001432      MOV      #36,TSTNM           ;MOVE #36 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

014604 005737 001436      TST      RH70                ;TEST FLAG FOR RH70 CONTROLLER
014610 001402                    BEQ      64$                    ;IF FLAG = 1, THIS TEST IS SKIPPED
014612 000137 014752      JMP      TST37                ;JUMP TO NEXT TEST
014616                                     64$:
75 014616 004737 041460      JSR      PC,CLDISK            ;GIVE INIT & SETUP REGISTER CORRES
76
77 014622 052712 177000      BIS      #177000,(R2)         ;LOAD RHCS2
78 014626 010246                    MOV      R2,-(SP)              ;GETTING READY FOR ODD BYTE
79 014630 005216                    INC      (SP)                  ;SP NOW HAS ODD BYTE FOR RHCS2
80 014632 105036                    CLRB    @(SP)+                 ;CLERR RHCS2 ODD BYTE
81 014634 013746 001374      MOV      UNIT,-(SP)           ;GET UNIT NO.
82 014640 052716 000100      BIS      #IR,(SP)             ;INPUT READY AS IT IS SET
83 014644 011237 001126      MOV      @R2,$BDDAT           ;TEST DATA
84 014650 022637 001126      CMP      (SP)+,$BDDAT         ;COMPARE TO SEE THAT
85                                     ;"CLRB" DID CLEAR
86 014654 001411                    BEQ      1$                    ;GAVE WRONG RESULTS
87 014656 013737 001374 001124      MOV      UNIT,$GDDAT          ;GOOD DATA
88 014664 052737 000100 001124      BIS      #IR,$GDDAT           ;FAILING REGISTER RHCS2
89 014672 010237 041130      MOV      R2,REGADR
90 014676 104001                    EMT      1
91
92 014700 013746 001374      1$:     MOV      UNIT,-(SP)         ;HAVE UPE AND MPE IN RHCS2
93 014704 052716 000010      BIS      #BAI,(SP)            ;BESIDES UNIT SELECT
94 014710 052712 020000      BIS      #UPE,@R2            ;MOVE INTO EVEN BYTE OF RHCS2
95
96 014714 112612                    MOVB    (SP)+,@R2
97 014716 013746 001374      MOV      UNIT,-(SP)
98 014722 052716 020110      BIS      #UPE!IR!BAI,(SP)
99 014726 011637 001124      MOV      (SP),$GDDAT          ;GOOD DATA
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
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

.SBTTL DCL COMMAND TESTS

:FOUR GENERAL REGISTERS WILL BE RESERVED FOR HARDWARE
:R1=RHCS1 CONTROL AND STATUS1
:R2=RHCS2 CONTROL AND STATUS2
:R3=RHDS1 DRIVE STATUS 1
:R4=RHER1 ERROR REGISTER1

:WHENEVER ANY OTHER USE IS MADE OF THESE REGISTERS
:APPROPRIATE SAVING MUST BE DONE

:ERROR REGISTER #01 (RHER1) TEST
:BIT #1 (ILLEGAL REGISTER) CANNOT BE TESTED ON PDP11 THIS BIT
:IS FOR PDP10 USE ONLY

*TEST 41 TEST ILF BIT #0 IN REG. RHER1
*ALL 3 ILLEGAL FUNCTION CODES SHOULD SET - ATA,ERR,ILF - AND ARE TESTED
*A GO WITHOUT CLEARING ILF ERR SHOULD SET - MXF,DLT,TRE - BITS AND THEY ARE ALSO TESTED

```
TST41: SCOPE
MOV #STACK,SP          :RESET STACK
MOV #41,TSTNM          :MOVE #41 TO TEST NUMBER
JSR PC,CLDISK          :CLEAR REGISTERS
MOV #DMD,@RHMR        :SET DIAGNOSTIC MODE
CLR TMPILL             :GET READY TO MAKE ILLEGAL FUNCTION
1$: MOV #FUTABL,R0      :LOAD FUNCTION CODE TABLE START
MOV #17,R5             :COUNTER (16 GOOD FUNCTIONS)
2$: CMP TMPILL,(R0)+   :IS THIS A LEGAL FUNCTION CODE?
BNE 3$                 :NO - DECR. FUNCT. CODE CTR
ADD #2,TMPILL          :YES MAKE NEXT FUNCTION CODE
BR 1$                  :TEST NEXT FUNCTION CODE
3$: DEC R5              :MAKE NEXT CODE IF 1ST 16
                           :LEGAL FUNCTIONS NOT DONE
BNE 2$                 :BRANCH IF 16 NOT COMPLETE
BIT #100,TMPILL        :ALL BITS UP TO BIT #5 COMPARED?
BNE 12$                :YES - EXIT
MOV TMPILL,ILLEGL;NO  : - TEST THE ILLEGAL FUNCTION
ADD #2,TMPILL          :TEST NEW FUNCTION CODE NEXT TIME
4$: JSR PC,CLDISK      :SET DIAGNOSTIC MODE
MOV #DMD,@RHMR        :ILLEGAL FUNCTION RHCS1
MOV ILLEGL,@R1        :ERROR RETURN POINT
MOV #4$, $LPERR       :SAVE REGISTERS
JSR PC,PUTREG         :THERE SHOULD NOT BE ANY ERROR YET
TST ER1               :CONTINUE IF RHER1 STILL = 0
BEQ 5$                :FAILING REGISTER ADDRESS RHER1
MOV R4,$BDADR
EMT 11
```

:HAS BEEN MOVED INTO RHCS1
:NO ERRORS SHOULD SHOW TILL
:GO IS SET RHER1 SHOULD BE
:ALL ZEROS

```

57
58 015376 052711 000001      5$:  BIS    #GO, @R1      ;GO IN RHCS1
59 015402 004737 041064      JSR    PC,PUTREG    ;SAVE REGISTERS
60 015406 022737 000001 001316  CMP    #ILF, ER1    ;ILLEGAL FUNCTION BIT SHOULD BE SET
61 015414 001403              BEQ    6$           ;IT IS - CONTINUE
62 015416 010437 001122      MOV    R4,$BDADR    ;FAILING REGISTER ADDRESS RHER1
63 015422 104011              EMT    11
64
65                          ;SET ON AN ILLEGAL FUNCTION
66                          ;EXECUTION, THE ILLEGAL FUNCTION
67                          ;BEING EXECUTED IS IN RHCS1
68 015424 013746 001336      6$:  MOV    DS1,-(SP)    ;GET RHDS1
69 015430 042716 001000      BIC    #PROG,(SP)   ;MASK PROG
70 015434 022726 140700      CMP    #ATA!ERR!VV!DPR!DRY,(SP)+
71                          ;ATTENTION (BIT 15)
72                          ;VOLUME VALID (BIT 6)
73                          ;COMPOSIT ERROR (BIT 14)
74                          ;DEVICE READY (BIT 7) SHOULD
75                          ;BE SET ON RHDS1
76 015440 001404              BEQ    7$           ;THEY ARE - CONTINUE
77 015442 013737 001262 001122  MOV    RHDS1,$BDADR ;FAILING REGISTER ADDRESS RHDS1
78 015450 104011              EMT    11
79
80                          ;WITH AN ILLEGAL FUNCTION
81                          ;ATTENTION (BIT 15)
82                          ;COMPOSIT ERROR (BIT 14)
83                          ;MEDIUM ON LINE (BIT 12)
84                          ;DEVICE READY (BIT 7)
85 015452 004737 043720      7$:  JSR    PC,MIDDLE  ;GIVE A WRITE HEADER AND
86                          ;DATA COMMAND WITHOUT
87                          ;CLEARING THE ERRORS
88                          ;USING 'MIDDLE' SO THAT
89                          ;IT WILL COME BACK BEFORE
90                          ;THE END TO FIND OUT ITS
91                          ;STATE
92 015456 010237 015464      MOV    R2,10$      ;MOVE RHCS2 ADDRESS
93 015462 104415              WAT
94 015454 000000      10$: .WORD 0          ;WAIT FOR 'MXF' BIT
95 015466 001000              MXF              ;ADDRESS OF RHCS2
96 015470 004737 041064      11$: JSR    PC,PUTREG    ;SAVE REGISTERS
97
98 015474 032737 040000 001314  BIT    #TRE,CS1    ;TRANSFER ERROR (BIT 14) RHCS1 - 'TRE'
99                          ;SHOULD SET DUE TO 'MXF'
100 015502 001003              BNE    13$        ;IT IS - CONTINUE
101 015504 010137 001122      MOV    R1,$BDADR    ;FAILING REGISTER RHCS1
102 015510 104011              EMT    11
103
104                          ;SHOULD BE SET DUE TO 'MXF'
105                          ;LOCAL SCOPE RETURN POINT
106 015512 000660      13$: BR    1$       ;GO BACK & TEST NEXT FUNCTION CODE
107
108 015514 000240      12$: NOP
109
110
111
112
113
114
115
116
;*****
;*TEST 42      READ IN PRESET
;*ALL POSSIBLE REGISTERS WILL BE FILLED WITH ONES
;*THE REGISTER CONTENTS WILL BE SAVED IN REINTO BUFFER
    
```

;*THE READ IN PRESET COMMAND WILL BE GIVEN
 ;*ALL REGISTERS WILL BE CHECKED
 ;*****

```

117 015516 000004
118 015520 012706 001100
119 015524 012737 000042 001432
120 015532 004737 041460
121
122
123
124 015536 012777 177777 163466
125 015544 012777 177777 163462
126 015552 012777 017437 163464
127 015560 012777 016377 163462
128 015566 012777 000777 163456
129 015574 012746 001400
130 015600 053716 001502
131 015604 012677 163430
132 015610 012777 000001 163442
133
134
135 015616 004037 042152
136 015622 001232
137 015624 002554
138 015626 000021
139
140
141 015630 052777 000001 163402
142
143
144 015636 005037 002566
145 015642 042737 016000 002572
146
147 015650 052737 000100 002572
148 015656 005037 002574
149
150
151
152
153 015662 004037 042152
154 015666 001232
155 015670 001510
156 015672 000021
157
158
159
160
161 015674 113737 002601 001535
162
163
164
165
166 015702 004037 042354
167 015706 002554
168 015710 001510
169 015712 000021
  
```

```

TST42: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #42,TSTNM ;MOVE #42 TO TEST NUMBER
JSR PC,CLDISK ;INIT AND SET GENERAL REGISTERS
;FILL ALL POSSIBLE BITS WITH ONES
MOV #177777,@RHWC ;WORD COUNT REGISTER GETS 177777
MOV #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777
MOV #17437,@RHDST ;DESIRED SECTOR TRACK GETS 17437
MOV #16377,@RHOF ;OFFSET REGISTER GETS 16277
MOV #777,@RHCA ;DESIRED CYLINDER GETS 777
MOV #A16!A17,-(SP) ;GET BIT 9 AND 8
BIS READIN,(SP)
MOV (SP)+,@RHCS1 ;FILL READ IN PRESET IN RHCS1
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
;THE REGISTERS WILL BE SAVED IN REINTO BUFFER
JSR R0,SAVER ;SAVE
RHCW ;FROM
REINTO ;TO
17. ;NUMBER SAVED
;GIVE READ IN PRESET COMMAND
BIS #GO,@RHCS1 ;INCLUDE GO TO READ IN PRESET
;NOW SAVED REGISTERS WILL BE CHANGED TO EXPECTED VALUE
CLR REINTO+12 ;CLEAR SAVED RHDST
BIC #FMT22!HCI!ECI,REINTO+16 ;CLEAR FMT22,HCI,ECI IN
;SAVED RHOF
BIS #VV,REINTO+16 ;SET VV IN SAVED RHOF
CLR REINTO+20 ;CLEAR SAVED RHCA
;AFTER A READ IN PRESET COMMAND
;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
JSR R0,SAVER ;SAVE
RHCW ;FROM
WRFROM ;TO
17. ;NUMBER OF REGISTERS SAVED
;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
MOVB REINTO+25,WRFROM+25;SAVE UPPER RHAS
;COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
;WITH AFTER COMMAND
JSR R0,COMPAR ;COMPARE
REINTO ;GOOD BUFFER
WRFROM ;TEST BUFFER
17. ;NUMBER OF REGISTERS
  
```

```

T42
170 015714 015722          1$          ;RETURN FOR ERROR
171 015716 015722          1$          ;SAME
172 015720 015742          2$          ;RETURN FOR GOOD COMPARISON
173
174 015722 013705 046426  1$:  MOV     ERWORD,R5      ;GETTING READY TO INDEX
175 015726 060505          ADD     R5,R5          ;DOUBLE ERROR WORD
176 015730 016537 001230 041130  MOV     RHWC-2(R5),REGADR ;FAILING REG. ADDRESS
177 015736 104001          EMT     1
178
179 015740 000207          RTS     PC              ;REGISTER CHANGE
180
181 015742          2$:          ;NO ERRORS
182
189
;*****
;*TEST 43      NO OPERATION FUNCTION TEST
;*ALL POSSIBLE REGISTERS ARE CLEARED THEN A 'NOP'='C
;*IS GIVEN NO CHANGE SHOULD HAPPEN
;*ALL POSSIBLE REGISTERS ARE FILLED WITH ONES THEN A 'NOP'
;*IS GIVEN NO CHANGE SHOULD HAPPEN
;*****
190 015742 000004          TST43: SCOPE
191 015744 012737 000043 001432  MOV     #43,TSTNM      ;MOVE #43 TO TEST NUMBER
192
193          ;START WITH CLR IN RHCS2 (BIT5)
194 015752 004737 041460          JSR     PC,CLDISK      ;CLEAR ALL POSSIBLE BITS
195 015756 012777 000001 163274  MOV     #DMD,@RHMR     ;SET DIAGNOSTIC MODE
196 015764 013711 001442          MOV     NOPERA,@R1     ;PUT NOP OPERATION=0 IN RHCS1
197 015770 012700 001232          MOV     #RHWC,R0       ;STARTING ADDRESS OF REG
198 015774 012703 001306          MOV     #WC,R3         ;STARTING ADDRESS OF WHERE SAVED
199 016000 012702 000021          MOV     #RHEC2-RHWC+2/2,R2 ;NUMBER OF REGISTERS
200 016004 013023          1$:  MOV     @R0+,(R3)+    ;SAVE HARDWARE REG
201 016006 005302          DEC     R2              ;COUNT
202 016010 001375          BNE     1$             ;BRANCH IF NOT COMPLETE
203 016012 013737 001262 016032  MOV     RHDS1,2$       ;GET ADDRESS OF DRIVE STATUS
204 016020 010137 016040          MOV     R1,3$         ;GET ADDRESS OF RHCS1
205 016024 052711 000001          BIS     #GO,@R1       ;GO TO RHCS1
206 016030 104415          WAT          ;WAIT FOR DRY IN RHDS1
207 016032 000000          2$:  .WORD  0           ;ADDRESS OF DRIVE STATUS RHDS1
208 016034 000200          DRY          ;DRY WILL BE WAITED ON
209 016036 104415          WAT          ;WAIT FOR RDY IN RHCS1
210 016040 000000          3$:  .WORD  0           ;ADDRESS OF RHCS1 PUT HERE BY AN
211
212 016042 000200          RDY          ;EARLIER MOV
213
214          ;AFTER A NO OP COMMAND
215          ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
216
217 016044 004037 042152          JSR     R0,SAVER      ;SAVE
218 016050 001232          RHWC     ;FROM
219 016052 001510          WRFROM   ;TO
220 016054 000021          17.      ;NUMBER OF REGISTERS SAVED
221
222          ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
223          ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
224          ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
225 016056 113737 001333 001535  MOVB    AS+1,WRFROM+25;SAVE UPPER RHAS

```

```

226
227
228           ;COMPARE REGISTERS BEFORE NO OP COMMAND
229           ;WITH AFTER COMMAND
230
231 016064 004037 042354           JSR    R0,COMPARE           ;COMPARE
232 016070 001306                   WC                       ;GOOD BUFFER
233 016072 001510                   WRFROM                    ;TEST BUFFER
234 016074 000021                   17.                       ;NUMBER OF REGISTERS
235 016076 016104                   4$                        ;RETURN FOR ERROR
236 016100 016104                   4$                        ;SAME
237 016102 016124                   5$                        ;RETURN FOR GOOD COMPARISON
238
239 016104 013705 046426           4$:  MOV    ERWORD,R5           ;GETTING READY TO INDEX
240 016110 060505                   ADD    R5,R5              ;DOUBLE ERROR WORD
241 016112 016537 001230 041130   MOV    RHWC-2(R5),REGADR ;FAILING REG. ADDRESS
242 016120 104001                   EMT    1
243                                     ;REGISTER CHANGE
244 016122 000207                   RTS    PC                 ;RETURN FOR FURTHER COMPARISONS
245
246 016124           5$:                                     ;NO ERRORS
247
248
249
250 016124 012737 016132 001110   MOV    #14$,$LPERR        ;SET SCOPE LOOP TO 14$
251 016132 004737 041460           14$: JSR    PC,CLDISK        ;INIT LAST ALL ZERO TEST
252 016136 012777 000001 163114   MOV    #DMD,@RHMR         ;SET DIAGNOSTIC MODE
253
254
255           ;NOW START WITH ALL ONES IN ALL POSSIBLE REGISTERS
256
257 016144 012700 001232           MOV    #RHWC,R0           ;ADDRESS OF FIRST REGISTER
258 016150 012705 000021           MOV    #RHEC2-RHWC+2/2,R5 ;NO. OF REGISTERS
259 016154 012730 177676           6$:  MOV    #177676,@(R0)+ ;FILL WITH ALL ONES
260 016160 013777 001374 163050   MOV    UNIT,@RHCS2        ;REINSTATE UNIT NUMBER UNDER TEST
261                                     ;KEEP INTERRUPT DISABLED
262 016166 005305                   DEC    R5                 ;COUNT
263 016170 001371                   BNE    6$                 ;BRANCH IF INCOMPLETE
264 016172 013711 001442           MOV    NOPERA,@R1         ;PUT NOP OPERATION =0 IN RHCS1
265 016176 012700 001232           MOV    #RHWC,R0           ;STARTING ADDRESS OF REG
266 016202 012703 001306           MOV    #WC,R3             ;STARTING ADDRESS OF WHERE SAVED
267 016206 012702 000021           MOV    #RHEC2-RHWC+2/2,R2 ;NUMBER OF REGISTERS
268 016212 013023           7$:  MOV    @(R0)+,(R3)+     ;SAVE HARDWARE REG
269 016214 005302                   DEC    R2                 ;COUNT
270 016216 001375                   BNE    7$                 ;BRANCH IF NOT COMPLETE
271 016220 013737 001262 016240   MOV    RHDS1,10$         ;GET ADDRESS OF DRIVE STATUS
272 016226 010137 016246           MOV    R1,11$           ;GET ADDRESS OF RHCS1
273 016232 052711 000001           BIS    #GO,@R1           ;GO TO RHCS1
274 016236 104415                   WAT                               ;WAIT FOR DRY IN RHDS1
275 016240 000000           10$: .WORD 0              ;ADDRESS OF DRIVE STATUS RHDS1
276 016242 000200                   DRY                               ;DRY WILL BE WAITED ON
277 016244 104415                   WAT                               ;WAIT FOR RDY IN RHCS1
278 016246 000000           11$: .WORD 0              ;ADDRESS OF RHCS1 PUT HERE BY AN
279                                     ;EARLIER MOV.
280 016250 000200                   RDY                               ;RDY WILL BE WAITED ON
281
282           ;AFTER A NO OP COMMAND

```



```

283                                     ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
284
285 016252 004037 042152                JSR    R0,SAVER          ;SAVE
286 016256 001232                        RHWC          ;FROM
287 016260 001510                        WRFROM       ;TO
288 016262 000021                        17.          ;NUMBER OF REGISTERS SAVED
289
290                                     ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
291                                     ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
292                                     ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
293 016264 113737 001333 001535         MOVB   AS+1,WRFROM+25;SAVE UPPER RHAS
294
295
296                                     ;COMPARE REGISTERS BEFORE NO OP COMMAND
297                                     ;WITH AFTER COMMAND
298
299 016272 004037 042354                JSR    R0,COMPAR        ;COMPARE
300 016276 001306                        WC          ;GOOD BUFFER
301 016300 001510                        WRFROM       ;TEST BUFFER
302 016302 000021                        17.          ;NUMBER OF REGISTERS
303 016304 016312                        12$         ;RETURN FOR ERROR
304 016306 016312                        12$         ;SAME
305 016310 016332                        13$         ;RETURN FOR GOOD COMPARISON
306
307 016312 013705 046426                12$:  MOV   ERWORD,R5    ;GETTING READY TO INDEX
308 016316 060505                        ADD   R5,R5        ;DOUBLE ERROR WORD
309 016320 016537 001230 041130         MOV   RHWC-2(R5),REGADR ;FAILING REG. ADDRESS
310 016326 104001                        EMT   1
311                                     ;REGISTER CHANGE
312 016330 000207                        RTS   PC          ;RETURN FOR FURTHER COMPARISONS
313
314 016332                                13$:          ;NO ERRORS
315
323                                     ;*****
324                                     ;*TEST 44 DRIVE CLEAR
325                                     ;*ALL WRITE BITS OF ALL REGISTERS EXCEPT RHDB ARE FILLED WITH
326                                     ;*ONES EXCEPT FOR BIT #0 AND BIT #6 WHICH ARE 'GO' AND
327                                     ;*'ENABLE INTERRUPT' BITS
328                                     ;*THEN A DRIVE CLEAR IS PERFORMED
329                                     ;*THEN ALL REGISTERS EXCEPT RHDB ARE CHECKED
330                                     ;*****
331 016332 000004                                TST44: SCOPE
332 016334 012706 001100                        MOV   #STACK,SP    ;RESET STACK
333 016340 012737 000044 001432         MOV   #44,TSTNM    ;MOVE #44 TO TEST NUMBER
334 016346 004737 041460                        JSR   PC,CLDISK    ;SET REGISTERS AND CLEAR
335
336                                     ;FILL ALL POSSIBLE BITS WITH ONES
337
338 016352 012777 177777 162650         MOV   #177777,@RHDB ;BUS ADDRESS REGISTER GETS 177777
339 016360 012777 177777 162644         MOV   #177777,@RHWC ;WORD COUNT REGISTER GETS 177777
340 016366 012777 177777 162640         MOV   #177777,@RHBA ;BUS ADDRESS REGISTER GETS 177777
341 016374 052777 157010 162634         BIS   #157010,@RHCS2 ;CONTROL AND STATUS 2 GETS 157010
342 016402 012777 001476 162630         MOV   #1476,@RHCS1  ;CONTROL AND STATUS REGISTER/GETS 1476
343 016410 012777 177777 162624         MOV   #177777,@RHER1 ;ERROR REGISTER1 GETS 177777
344 016416 012777 017437 162620         MOV   #17437,@RHDST ;DESIRED SECTOR TRACK
345 016424 012777 177777 162614         MOV   #177777,@RHER2 ;ERROR REGISTER 2
346 016432 012777 016277 162610         MOV   #16277,@RHOF  ;OFFSET REGISTER

```

```

339 016440 012777 177777 162604      MOV      #177777,@RHCA      ;DESIRED CYLINDER
340 016446 012777 177777 162600      MOV      #177777,@RHER3    ;ERROR REGISTER 3
341 016454 012777 000001 162576      MOV      #DMD,@RHMR       ;MAINTENANCE REGISTER
342 016462 012777 177777 162570      MOV      #177777,@RHMR     ;MAINTENANCE REGISTER
343
344
345      ;THIS SETS BITS FOR ALL PRESENT DRIVES
346
347 016470 013700 001420      MOV      TOTALAT,R0        ;GET DRIVE PRESENT
348 016474 005012      CLR      @R2               ;CLEAR RHCS2 AND CARRY BIT
349 016476 012705 000010      MOV      #8.,R5           ;COUNTER
350 016502 006000      30$:    ROR      R0          ;GET BIT INTO CARRY
351 016504 103002      BCC     31$               ;BRANCH IF NO UNIT ON THIS BIT
352 016506 012714 177777      MOV      #-1,@R4          ;MOVE INTO ERROR REGISTER TO SET ATA
353 016512 005212      31$:    INC      @R2        ;INCREMENT RHCS2 - UNIT NO.
354 016514 005305      DEC     R5               ;COUNT
355 016516 001401      BEQ     27$               ;BRANCH IF 8 DONE
356 016520 000770      BR      30$              ;CONTINUE THIS ROUTINE
357 016522 013746 001374      27$:    MOV      UNIT,-(SP)   ;
358 016526 052716 157010      BIS     #157010,(SP)      ;REINSTATE SET BITS
359 016532 012612      MOV     (SP)+,@R2        ;
360
361
362 016534 012777 000001 162516      MOV      #DMD,@RHMR       ;SET DMD
363 016542 013711 001450      MOV     DCLEAR,@R1       ;DRIVE CLEAR = 10 INTO RHCS1
364 016546 052711 000001      BIS     #GO,@R1          ;GO
365 016552 012700 001230      MOV     #RHDB,R0         ;R0 CONTAINS ADDR. OF ADDR. OF REG.
366
367
368      ;DATA BUFFER REGISTER
369
370      016556 012737 177777 001124 28$:    MOV      #177777,$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
      016564 013037 001126      MOV     @(R0)+,$BDDAT     ;TEST DATA
      016570 022737 177777 001126      CMP     #177777,$BDDAT    ;COMPARE DATA
      016576 001402      BEQ     3$                ;BRANCH IF GOOD
      016600 004737 017436      JSR     PC,ERCLFC         ;JUMP TO ERROR FOR CLR (BIT 5)
      ;IN RHCS2
371
372
373      ;WORD COUNT REGISTER
374
375      016604 012737 177777 001124 3$:    MOV      #177777,$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
      016612 013037 001126      MOV     @(R0)+,$BDDAT     ;TEST DATA
      016616 022737 177777 001126      CMP     #177777,$BDDAT    ;COMPARE DATA
      016624 001402      BEQ     4$                ;BRANCH IF GOOD
      016626 004737 017436      JSR     PC,ERCLFC         ;JUMP TO ERROR FOR CLR (BIT 5)
      ;IN RHCS2
376
377
378      ;BUS ADDRESS REGISTER
379
380      016632 012737 177776 001124 4$:    MOV      #177776,$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
      016640 013037 001126      MOV     @(R0)+,$BDDAT     ;TEST DATA
      016644 022737 177776 001126      CMP     #177776,$BDDAT    ;COMPARE DATA
  
```

016652 001402 BEQ 5\$:BRANCH IF GOOD
016654 004737 017436 JSR PC,ERCLFC :JUMP TO ERROR FOR CLR (BIT 5)
:IN RHCS2

381
382
383
384
385
386

:CONTROL AND STATUS 2 REGISTER

016660 012746 000110 5\$: MOV #110,-(SP) :INCLUDE IR
016664 053716 001374 BIS UNIT,(SP) :SET UNIT NO.
016670 012637 001124 MOV (SP)+,\$GDDAT :GOOD DATA FOR TYPE OUT
016674 013037 001126 MOV @ (R0)+,\$BDDAT :TEST DATA
016700 023737 001124 001126 CMP \$GDDAT,\$BDDAT :COMPARE DATA
016706 001402 BEQ 6\$:BRANCH IF GOOD
016710 004737 017436 JSR PC,ERCLFC :JUMP TO ERROR FOR CLR (BIT 5)
:IN RHCS2

387
388
389
390

:CONTROL AND STATUS 1 REGISTER

016714 005737 001400 6\$: TST NUNIT :ARE THERE MORE THAN ONE UNIT
016720 001404 BEQ 32\$:BRANCH IF ONLY ONE UNIT
016722 012737 104210 001124 MOV #104210,\$GDDAT :GOOD DATA
016730 000403 BR 33\$
016732 012737 004210 001124 32\$: MOV #4210,\$GDDAT :GOOD DATA
016740 013037 001126 33\$: MOV @ (R0)+,\$BDDAT :TEST DATA
016744 023737 001124 001126 CMP \$GDDAT,\$BDDAT :COMPARE DATA
016752 001402 BEQ 7\$:BRANCH IF GOOD
016754 004737 017436 JSR PC,ERCLFC :JUMP TO ERROR FOR CLR BIT 5
:IN RHCS2

401
402
403
404
405

:ERROR 1 REGISTER

016760 012737 000000 001124 7\$: MOV #0,\$GDDAT :GOOD DATA FOR ERROR TYPEOUT
016766 013037 001126 MOV @ (R0)+,\$BDDAT :TEST DATA
016772 022737 000000 001126 CMP #0,\$BDDAT :COMPARE DATA
017000 001402 BEQ 10\$:BRANCH IF GOOD
017002 004737 017436 JSR PC,ERCLFC :JUMP TO ERROR FOR CLR (BIT 5)
:IN RHCS2

406
407
408
409

:DESIRED SECTOR/TRACK REGISTER

017006 012737 017437 001124 10\$: MOV #17437,\$GDDAT :GOOD DATA FOR ERROR TYPEOUT
017014 013037 001126 MOV @ (R0)+,\$BDDAT :TEST DATA
017020 022737 017437 001126 CMP #17437,\$BDDAT :COMPARE DATA
017026 001402 BEQ 11\$:BRANCH IF GOOD
017030 004737 017436 JSR PC,ERCLFC :JUMP TO ERROR FOR CLR (BIT 5)
:IN RHCS2

410
411
412
413

:ERROR 2 REGISTER

017034 012737 000000 001124 11\$: MOV #0,\$GDDAT :GOOD DATA FOR ERROR TYPEOUT


```

442 017246 012737 000700 001124 17$:  MOV    #700,$GDDAT    ;GOOD DATA FOR PRINTOUT
443 017254 013046                MOV    @ (R0)+,-(SP)    ;GET RHDS1
444 017256 011637 001126                MOV    (SP),$BDDAT    ;TEST DATA
445 017262 042716 001000                BIC    #PROG,(SP)    ;CLEAR PROG BIT
446 017266 022726 000700                CMP    #700,(SP)+    ;COMPARE DATA
447 017272 001402                BEQ    20$           ;BRANCH IF GOOD
448 017274 004737 017436                JSR    PC,ERCLFC    ;JUMP TO ERROR FOR DRIVE CLEAR
449
450                ;DRIVE TYPE
451
452 017300 013737 001410 001124 20$:  MOV    SAVDT,$GDDAT    ;GOOD DATA FOR ERROR TYPEOUT
017306 013037 001126                MOV    @ (R0)+,$BDDAT ;TEST DATA
017312 023737 001410 001126                CMP    SAVDT,$BDDAT  ;COMPARE DATA
017320 001402                BEQ    21$           ;BRANCH IF GOOD
017322 004737 017436                JSR    PC,ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
453
454                ;SERIAL NUMBER REGISTER
455
456
457 017326 013737 001412 001124 21$:  MOV    SAVSN,$GDDAT   ;GOOD DATA FOR ERROR TYPEOUT
017334 013037 001126                MOV    @ (R0)+,$BDDAT ;TEST DATA
017340 023737 001412 001126                CMP    SAVSN,$BDDAT  ;COMPARE DATA
017346 001402                BEQ    22$           ;BRANCH IF GOOD
017350 004737 017436                JSR    PC,ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
458
459                ;ECC1 POSITION
460
461 017354 012737 000000 001124 22$:  MOV    #0,$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
017362 013037 001126                MOV    @ (R0)+,$BDDAT ;TEST DATA
017366 022737 000000 001126                CMP    #0,$BDDAT     ;COMPARE DATA
017374 001402                BEQ    23$           ;BRANCH IF GOOD
017376 004737 017436                JSR    PC,ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
462
463                ;ECC2 PATTERN
464
465 017402 012737 000000 001124 23$:  MOV    #0,$GDDAT     ;GOOD DATA FOR ERROR TYPEOUT
017410 013037 001126                MOV    @ (R0)+,$BDDAT ;TEST DATA
017414 022737 000000 001126                CMP    #0,$BDDAT     ;COMPARE DATA
017422 001402                BEQ    24$           ;BRANCH IF GOOD
017424 004737 017436                JSR    PC,ERCLFC    ;JUMP TO ERROR FOR CLR (BIT 5)
                                ;IN RHCS2
466
467                ;LOOK-AHEAD REGISTER
468
469
470 017430 005720                24$:  TST    (R0)+        ;AS THE LOOK-AHEAD REG. CANNOT BE PREDICTED
471                                ;IT IS NOT CHECKED AFTER AN INIT
472
473                ;CURRENT CYLINDER ADDRESS REGISTER
474

```

```

475 017432 005720      25$:  TST      (R0)+      ;AS THE CURRENT CYL REG. CANNOT BE PREDICTED
476                                     ;AFTER AN INIT IT IS NOT CHECKED
477
478 017434 006413      26$:  BR        TST45      ;BRANCH OVER JSR
479
480 017436 014037 041130  ERCLFC: MOV     -(R0),REGADR ;FAILING REGISTER ADDRESS
481 017442 104001      EMT        1
482                                     ;NOT CLEAR APPROPRIATE BITS
483                                     ;OR CLEARED EXTRA BITS
484 017444 005720      TST      (R0)+      ;UNDO -(R0) FOR ERROR
485 017446 000207      RTS        PC        ;RETURN TO ABOVE PROGRAM
486                                     ;OR CLEARED EXTRA BITS
487 017450 005720      TST      (R0)+      ;UNDO -(R0) FOR BAD DATA
488 017452 004737 041514 JSR        PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
      017456 104401 062726 TYPE      ,CPHALT  ;AND THAT NO OTHERS = 1. CANNOT CON-
                                     ;TINUE TESTING IF BOTH AREN'T TRUE
      017462 000000      HALT
489
496

```

```

:*****
:*TEST 45      SEEK COMMAND TEST
:*THE SEEK COMMAND WILL BE LOADED INTO RHCS1 WITH GO
:*THEN ALL REGISTERS WILL BE CHECKED
:*RH CLEAR WILL BE GIVEN
:*THEN ALL REGISTERS WILL BE CHECKED
:*****

```

```

497 017464 000004      TST45: SCOPE
498 017466 012706 001100  MOV     #STACK,SP      ;RESET STACK
499 017472 012737 000045 001432  MOV     #45,TSTNM      ;MOVE #45 TO TEST NUMBER
500 017500 004737 041460      JSR     PC,CLDISK      ;INIT AND SET UP GENERAL REG. CORRES.
501 017504 012777 000001 161546  MOV     #DMD,@RHMR     ;AND UNIT NUMBER
502                                     ;SET DIAGNOSTIC MODE BIT
503                                     ;THIS ENABLES COMMANDS WITHOUT MOL
504 017512 005077 161526      CLR     @RHDST         ;AND HOLDS RHLA FROM MOVING
505 017516 013777 001472 161514  MOV     SEECOM,@RHCS1 ;MAKE DESIRED SECTOR TRACK LEGAL
506 017524 017746 161546      MOV     @RHCC,-(SP)   ;LOAD SEEK COMMAND INTO CONTROLLER
507                                     ;GET CURRENT CYLINDER
508
509
510
511

```

```

;FOLLOWING ARE TWO BLOCKS OF CODE TO LOAD RHCA WITH THE PROPER
;ADDRESS DEPENDING UPON WHETHER THE DRIVE IS AN RP06 OR RP04

```

```

512 017530 005737 001434      TST     RP06          ;MOVE DRIVE TYPE FLAG TO ITSELF TO TEST
513 017534 001416      BEQ     11$          ;TREAT THE DRIVE AS AN RP04
514
515
516 017536 022726 001456      CMP     #814.,(SP)+   ;TREAT THE DRIVE AS AN RP06
517 017542 001404      BEQ     9$           ;IS CURRENT CYLINDER SAME AS 814. ?
518 017544 012737 001456 001210  MOV     #814.,$TMP5   ;BRANCH IF YES TO MAKE RHCA = 813.
519 017552 000403      BR      10$          ;GET READY TO MAKE RHCA = 814.
520 017554 012737 001455 001210 9$:  MOV     #813.,$TMP5   ;FILL RHCA
521 017562 013777 001210 161462 10$: MOV     $TMP5,@RHCA   ;GET READY TO MAKE RHCA = 813.
522 017570 000415      BR      14$          ;MAKE DESIRED CYLINDER 814., OR 813.
523                                     ;SAVE REGISTERS
524
525 017572 022726 000632      11$:  CMP     #410.,(SP)+  ;TREAT THE DRIVE AS AN RP04
526 017576 001404      BEQ     12$          ;IS CURRENT CYLINDER SAME AS 410. ?
                                     ;BRANCH IF YES TO MAKE RHCA = 409.

```



```

584
585           ;NOW GIVE INIT AND GET GO AND PIP DOWN
586
587 017746 052712 000040      2$:   BIS   #CLR,@R2      ;RH INITILIZE
588 017752 013712 001374      MOV   UNIT,@R2      ;REINSTATE UNIT NUMBER
589 017756 012777 000001 161274  MOV   #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
590                                           ;THIS ENABLES COMMANDS WITHOUT MOL
591                                           ;AND HOLDS RHLA FROM MOVING
592
593           ;CHANGE REGISTERS TO EXPECTED VALUE
594
595 017764 042737 000001 002562  BIC   #GO,REINTO+6  ;SAVED RHCS1
596 017772 042737 020000 002604  BIC   #PIP,REINTO+30 ;SAVED RHDS1
597 020000 052737 000200 002604  BIS   #DRY,REINTO+30 ;SAVED RHDS1
598 020006 017737 161262 002616  MOV   @RHLA,REINTO+42;SAVED RHLA
599 020014 013737 001210 002620  MOV   $TMP5,REINTO+44 ;SAVED RHCC
600
601
602           ;AFTER INITILIZE SAVE REGISTERS SO THAT
603           ;COMPARES CAN BE DONE
604
605 020022 004037 042152      JSR   R0,SAVER      ;SAVE
606 020026 001232      RHCW      ;FROM
607 020030 001510      WRFROM     ;TO
608 020032 000023      19.        ;NUMBER OF REGISTERS SAVED
609
610
611           ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
612           ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
613           ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
614
615 020034 113737 002601 001535  MOVB  REINTO+25,WRFROM+25;SAVE UPPER RHAS
616
617
618           ;COMPARE REGISTERS AFTER INITIALIZE
619
620 020042 004037 042354      JSR   R0,COMPAR     ;COMPARE
621 020046 002554      REINTO     ;GOOD BUFFER
622 020050 001510      WRFROM     ;TEST BUFFER
623 020052 000023      19.        ;NUMBER OF REGISTERS TO BE
624                                           ;COMPARED
625 020054 020062      3$         ;RETURN POINT FOR ERROR
626 020056 020062      3$         ;SAME
627 020060 020102      4$         ;RETURN POINT FOR GOOD COMPARISON
628
629 020062 013705 046426      3$:   MOV   ERWORD,R5      ;GETTING READY TO INDEX
630 020066 060505      ADD   R5,R5        ;DOUBLE ERROR WORD
631 020070 016537 001230 041130  MOV   RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
632 020076 104001      EMT      1
633                                           ;CONTENTS AFTER GIVING AN
634                                           ;INITILIZE FOLLOWING A
635                                           ;SEEK COMMAND
636 020100 000207      RTS   PC        ;RETURN TO COMPARISON
637
638           4$:
639
640 020102 004737 041460      JSR   PC,CLDISK    ;INIT AND SET UP GENERAL REG.
  
```



```
698 ;NOW GIVE INIT AND GET GO AND PIP DOWN
699
700 020250 052712 000040 6$: BIS #CLR,@R2 ;RH INITILIZE
701 020254 013712 001374 MOV UNIT,@R2 ;REINSTATE UNIT NUMBER
702 020260 012777 000001 160772 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
703 ;THIS ENABLES COMMANDS WITHOUT MOL
704 ;AND HOLDS RHLA FROM MOVING
705
706 ;CHANGE REGISTERS TO EXPECTED VALUE
707
708 020266 042737 000001 002562 BIC #GO,REINTO+6 ;SAVED RHCS1
709 020274 042737 020000 002604 BIC #PIP,REINTO+30 ;SAVED RHDS1
710 020302 052737 000200 002604 BIS #DRY,REINTO+30 ;SAVED RHDS1
711 020310 017737 160760 002616 MOV @RHLA,REINTO+42 ;SAVED RHLA
712 020315 005037 002620 CLR REINTO+44 ;SAVED RHCC
713
714
715 ;AFTER INITIALIZE SAVE REGISTERS SO THAT
716 ;COMPARES CAN BE DONE
717
718 020322 004037 042152 JSR R0,SAVER ;SAVE
719 020326 001232 RHWC ;FROM
720 020330 001510 WRFROM ;TO
721 020332 000023 19. ;NUMBER OF REGISTERS SAVED
722
723 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
724 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
725 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
726 020334 113737 002601 001535 MOVB REINTO+25,WRFROM+25;SAVE UPPER RHAS
727
728
729 ;COMPARE REGISTERS AFTER INITIALIZE
730
731 020342 004037 042354 JSR R0,COMPAR ;COMPARE
732 020346 002554 REINTO ;GOOD BUFFER
733 020350 001510 WRFROM ;TEST BUFFER
734 020352 000023 19. ;NUMBER OF REGISTERS TO BE
735 ;COMPARED
736 020354 020362 7$ ;RETURN POINT FOR ERROR
737 020356 020362 7$ ;SAME
738 020360 020402 8$ ;RETURN POINT FOR GOOD COMPARISON
739
740 020362 013705 046426 7$: MOV ERWORD,R5 ;GETTING READY TO INDEX
741 020366 060505 ADD R5,R5 ;DOUBLE ERROR WORD
742 020370 016537 001230 041130 MOV RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
743 020376 104001 EMT 1
744 ; COMMAND
745 020400 000207 RTS PC ;RETURN TO COMPARISON
746 020402 8$: ;GOOD COMPARISON
747
753 ;*****
; *TEST 46 UNLOAD COMMAND TEST
; *THE UNLOAD COMMAND WILL BE LOADED INTO RHCS1 WITH GO
; *THEN ALL REGISTERS WILL BE CHECKED
; *RH CLEAR WILL BE GIVEN
; ******
020402 000004 ;TST46: SCOPE
```

```

754 020404 012706 001100      MOV    #STACK,SP      ;RESET STACK
755 020410 012737 000046 001432  MOV    #46,TS1NM      ;MOVE #46 TO TEST NUMBER
756 020416 004737 041460      JSR    PC,CLDISK      ;INIT AND SET UP GENERAL REG.
757                                ;AND UNIT NUMBER
758 020422 012777 000001 160630  MOV    #DMD,@RHMR     ;SET DIAGNOSTIC MODE BIT
759                                ;THIS ENABLES COMMANDS WITHOUT MOL
760                                ;AND HOLDS RHLA FROM MOVING
761
762 020430 013777 001444 160602  MOV    UNLOAD,@RHCS1  ;LOAD UNLOAD COMMAND INTO RH
763
764                                ;SAVE REGISTERS FOR COMPARISON AFTER GO
765
766 020436 004037 042152      JSR    R0,SAVER       ;SAVE
767 020442 001232              RHWC                  ;FROM
768 020444 002554              REINTO                ;TO
769 020446 000023              19.                  ;NUMBER OF REGISTERS SAVED
770
771                                ;GIVE GO TO UNLOAD COMMAND
772 020450 052777 000001 160562  BIS    #GO,@RHCS1    ;GO TO UNLOAD COMMAND
773
774                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
775 020456 052737 000001 002562  BIS    #GO,REINTO+6  ;SAVED RHCS1
776 020464 052737 020000 002604  BIS    #PIP,REINTO+30 ;SAVED RHDS1
777 020472 042737 000200 002604  BIC    #DRY,REINTO+30 ;SAVED RHDS1
778
779 020500 005737 001100      TST    $PASS          ;IS THIS FIRST PASS
780 020504 001053              BNE    5$             ;BRANCH IF NOT FIRST PASS
781 020506 032777 020000 160424  BIT    #SW13,@SWR     ;INHIBIT ERROR PRINT HIGH?
782 020514 001047              BNE    5$             ;BRANCH IF SW13 HIGH
783 020516 104401 020524      TYPE   ,65$          ;:TYPE ASCIZ STRING
784                                BR    64$             ;:GET OVER THE ASCIZ
785                                ;:65$: .ASCIZ <CRLF>/IF DRIVE CONNECTED "STAND BY" LAMP SHOULD BE LIT ON DRIVE #/
786                                ;:64$:
787 020622 013746 001374      MOV    UNIT,-(SP)    ;UNIT UNDER TEST
788 020626 104405              TYPDS
789 020630 104401 001223      TYPE   ,$CRLF        ;CR-LF
790
791                                ;AFTER GO HAS BEEN GIVEN TO UNLOAD COMMAND
792                                ;SAVED REGISTERS AGAIN SO THAT COMPARISONS CAN
793                                ;BE DONE
794
795 020634 004037 042152      5$: JSR    R0,SAVER     ;SAVE
796 020640 001232              RHWC                  ;FROM
797 020642 001510              WRFROM                ;TO
798 020644 000023              19.                  ;NUMBER OF REGISTERS SAVED
799
800                                ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
801                                ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
802                                ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
803 020646 113737 002601 001535  MOVB   REINTO+25,WRFROM+25;SAVE UPPER RHAS
804
805                                ;COMPARE REGISTERS BEFORE UNLOAD COMMAND
806                                ;WITH AFTER GO
807 020654 004037 042354      JSR    R0,COMPAR     ;COMPARE
808 020660 002554              REINTO                ;GOOD BUFFER

```

```

808 020662 001510          WRFROM          ;TEST BUFFER
809 020664 000023          19.              ;NUMBER
810 020666 020674          1$              ;RETURN FOR ERROR
811 020670 020674          1$              ;SAME
812 020672 020714          2$              ;RETURN FOR GOOD COMPARISON
813 020674 013705 046426  1$:  MOV  ERWORD,R5 ;GETTING READY TO INDEX
814 020700 060505          ADD  R5,R5   ;DOUBLE ERROR WORD
815 020702 016537 001230 041130  MOV  RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
816
817 020710 104001          EMT  1
818
819
820 020712 000207          RTS  PC      ;AFTER UNLOAD COMMAND
821
822
823 020714 052712 000040  2$:  BIS  #CLR,@R2 ;RH INITILIZE
824 020720 013712 001374  MOV  UNIT,@R2 ;REINSTATE UNIT NUMBER
825 020724 012777 000001 160326  MOV  #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
826
827
828
829
830 020732 042737 000C01 002562  ;CHANGE REGISTERS TO EXPECTED VALUE
831 020740 042737 020000 002604  BIC  #GO,REINTO+6 ;SAVED RHCS1
832 020746 052737 000200 002604  BIC  #PIP,REINTO+30 ;SAVED RHDS1
833 020754 017737 160314 002616  BIS  #DRY,REINTO+30 ;SAVED RHDS1
834
835
836
837
838 020762 004037 042152          JSR  R0,SAVER ;SAVE
839 020766 001232          RHWC          ;FROM
840 020770 001510          WRFROM        ;TO
841 020772 000023          19.          ;NUMBER OF REGISTERS SAVED
842
843
844
845
846
847 020774 113737 002601 001535  ;COMPARE REGISTERS AFTER INITIALIZE
848
849
850 021002 004037 042354          JSR  R0,COMPAR ;COMPARE
851 021006 002554          REINTO        ;GOOD BUFFER
852 021010 001510          WRFROM        ;TEST BUFFER
853 021012 000023          19.          ;NUMBER OF REGISTERS TO BE
854
855 021014 021022          3$          ;COMPARED
856 021016 021022          3$          ;RETURN POINT FOR ERROR
857 021020 021042          4$          ;SAME
858
859 021022 013705 046426  3$:  MOV  ERWORD,R5 ;GETTING READY TO INDEX
860 021026 060505          ADD  R5,R5   ;DOUBLE ERROR WORD
861 021030 016537 001230 041130  MOV  RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
862 021036 104001          EMT  1
863
864

```

```

865 021040 000207          RTS      PC          ;RETURN TO COMPARISON
866 021042          4$:          ;GOOD COMPARISON
867
868 021042 032777 000020 160070      BIT      #SW4,@SWR      ;TEST FOR NO OFFSET OR RTC
869 021050 001402          BEQ      6$          ;IF = 0, DO THE NEXT TWO TESTS
870 021052 000137 022030          JMP      TST51        ;SKIP THE NEXT TWO TESTS
871 021056          6$:          ;CONTINUE WITH NEXT TWO TESTS
872
879
;*****
;*TEST 47      OFFSET COMMAND TEST
;*THE OFFSET COMMAND WILL BE LOADED INTO RHCS1 WITH GO
;*THEN ALL REGISTERS WILL BE CHECKED
;*RH CLEAR WILL BE GIVEN
;*THEN ALL REGISTERS WILL BE CHECKED
;*****
880 021056 000004          TST47:  SCOPE
881 021060 012706 001100      MOV      #STACK,SP      ;RESET STACK
882 021064 012737 000047 001432      MOV      #47,TSTNM      ;MOVE #47 TO TEST NUMBER
883 021072 004737 041460      JSR      PC,CLDISK      ;INIT AND SET UP GENERAL REG.
884 021076 012777 000001 160154      MOV      #DMD,@RHMR      ;AND UNIT NUMBER
885          ;SET DIAGNOSTIC MODE BIT
886          ;THIS ENABLES COMMANDS WITHOUT MOL
887          ;AND HOLDS RHLA FROM MOVING
888
889 021104 052777 000004 160146      ;GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
890 021112 042777 000004 160140      BIS      #MINX,@RHMR      ;SET INDEX PULSE
891          BIC      #MINX,@RHMR      ;CLEAR INDEX
892
893          ;TO ENABLE LOOP ON THIS TEST THE POSITIONER HAS TO
894          ;BE BROUGHT TO CENTER LINE
895 021120 017777 160152 160124      MOV      @RHCC,@RHCA      ;SET DESIRED CYLINDER TO RHCC
896 021126 013711 001472          MOV      SEECOM,@R1      ;SEEK COMMAND TO RHCS1
897 021132 005211          INC      @R1          ;GO TO SEEK COMMAND
898
899          ;FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER OFF OFFSET POSITION
900 02:134 012700 000004          MOV      #4,R0          ;COUNTER
901 021140 012777 000011 160112      5$:      MOV      #MSTCK!DMD,@RHMR      ;SET SECTOR CLOCK
902 021146 012777 000001 160104      MOV      #DMD,@RHMR      ;RESET SECTOR CLOCK
903 021154 005300          DEC      R0          ;COUNT
904 021156 001370          BNE     5$          ;BRANCH IF NOT COMPLETE
905
906 021160 004737 041460          JSR      PC,CLDISK      ;INIT AND SET UP GENERAL REG.
907          ;AND UNIT NUMBER
908 021164 012777 000001 160066      MOV      #DMD,@RHMR      ;SET DIAGNOSTIC MODE BIT
909          ;THIS ENABLES COMMANDS WITHOUT MOL
910          ;AND HOLDS RHLA FROM MOVING
911
912 021172 013777 001474 160040      MOV      OFSETC,@RHCS1    ;LOAD AN OFFSET BIT
913 021200 012777 000001 160042      MOV      #OF25,@RHOF     ;SET AN OFFSET BIT
914
915          ;SAVE REGISTERS FOR COMPARISON AFTER GO
916 02:206 004037 042152          JSR      R0,SAVER      ;SAVE
917 021212 001232          RHW      ;FROM
918 021217 002554          REINTO   ;TO
919 021216 000025          19.      ;NUMBER OF REGISTERS SAVED
920

```

```

921 ;GIVE GO TO OFFSET COMMAND
922 021220 052777 000001 160012 BIS #GO,@RHCS1 ;GO TO OFFSET COMMAND
923
924 ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
925 021226 052737 000001 002562 BIS #GO,REINTO+6 ;SAVED RHCS1
926 021234 052737 020000 002604 BIS #PIP,REINTO+30 ;SAVED RHDS1
927 021242 042737 000200 002604 BIC #DRY,REINTO+30 ;SAVED RHDS1
928
929 ;AFTER GO HAS BEEN GIVEN TO OFFSET COMMAND
930 ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
931 ;BE DONE
932
933 021250 004037 042152 JSR R0,SAVER ;SAVE
934 021254 001232 RHWC ;FROM
935 021256 001510 WRFROM ;TO
936 021260 000023 19. ;NUMBER OF REGISTERS SAVED
937
938 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
939 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
940 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
941 021262 113737 002601 001535 MOVB REINTO+25,WRFROM+25;SAVE UPPER RHAS
942
943
944 ;COMPARE REGISTERS BEFORE OFFSET COMMAND
945 ;WITH AFTER GO
946
947 021270 004037 042354 JSR R0,COMPAR ;COMPARE
948 021274 002554 REINTO ;GOOD BUFFER
949 021276 001510 WRFROM ;TEST BUFFER
950 021300 000023 19. ;NUMBER
951 021302 021310 1$ ;RETURN FOR ERROR
952 021304 021310 1$ ;SAME
953 021306 021330 2$ ;RETURN FOR GOOD COMPARISON
954
955 021310 013705 046426 1$: MOV ERWORD,R5 ;GETTING READY TO INDEX
956 021314 060505 ADD R5,R5 ;DOUBLE ERROR WORD
957 021316 016537 001230 041130 MOV RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
958 021324 104001 EMT 1
959 ;AFTER OFFSET COMMAND
960 ;WITH GO IS GIVEN
961 021326 000207 RTS PC ;RETURN TO COMPARISON
962
963 ;NOW GIVE INIT AND GET ALL GO AND PIP DOWN
964
965 021330 052712 000040 2$: BIS #CLR,@R2 ;RH INITILIZE
966 021334 013712 001374 MOV UNIT,@R2 ;REINSTATE UNIT NUMBER
967 021340 012777 000001 157712 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
968 ;THIS ENABLES COMMANDS WITHOUT MOL
969 ;AND HOLDS RHLA FROM MOVING
970
971 ;CHANGE REGISTERS TO EXPECTED VALUE
972 021346 042737 000001 002562 BIC #GO,REINTO+6 ;SAVED RHCS1
973 021354 042737 000001 002572 BIC #OF25,REINTO+16;SAVED RHOF
974 021362 042737 020000 002604 BIC #PIP,REINTO+30 ;SAVED RHDS1
975 021370 052737 000200 002604 BIS #DRY,REINTO+30 ;SAVED RHDS1
976 021376 017737 157672 002616 MOV @RHLA,REINTO+42;SAVED RHLA
977
  
```

```
978 ;AFTER INITIALIZE SAVE REGISTERS SO THAT
979 ;COMPARES CAN BE DONE
980
981 021404 004037 042152 JSR R0,SAVER ;SAVE
982 021410 001232 RHCW ;FROM
983 021412 001510 WRFROM ;TO
984 021414 000023 19. ;NUMBER OF REGISTERS SAVED
985
986 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
987 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
988 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
989 021416 113737 002601 001535 MOVB REINTO+25,WRFROM+25;SAVE UPPER RHAS
990
991 ;COMPARE REGISTERS AFTER INITIALIZE
992 021424 004037 042354 JSR R0,COMPAR ;COMPARE
993 021430 002554 REINTO ;GOOD BUFFER
994 021432 001510 WRFROM ;TEST BUFFER
995 021434 000023 19. ;NUMBER OF REGISTERS TO BE
996 ;COMPARED
997 021436 021444 3$ ;RETURN POINT FOR ERROR
998 021440 021444 3$ ;SAME
999 021442 021464 4$ ;RETURN POINT FOR GOOD COMPARISON
1000
1001 021444 013705 046426 3$: MOV ERWORD,R5 ;GETTING READY TO INDEX
1002 021450 060505 ADD R5,R5 ;DOUBLE ERROR WORD
1003 021452 016537 001230 041130 MOV RHCW-2(R5),REGADR ;FAILING REGISTER ADDRESS
1004 021460 104001 EMT 1
1005 ;CONTENTS AFTER GIVING AN
1006 ;INITIALIZE FOLLOWING A
1007 ;OFFSET COMMAND
1008 021462 000207 RTS PC ;RETURN TO COMPARISON
1009
1010 021464 4$: ;GOOD COMPARISON
1011
1018 ;:*****
; *TEST 50 RETURN TO CENTER LINE COMMAND TEST
; *THE RETURN TO CENTER LINE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
; *THEN ALL REGISTERS WILL BE CHECKED
; *RH CLEAR WILL BE GIVEN
; *THEN ALL REGISTERS WILL BE CHECKED
;:*****
1019 021464 000004 TST50: SCOPE
1020 021466 012706 001100 MOV #STACK,SP ;RESET STACK
1021 021472 012737 000050 001432 MOV #50,TSTNM ;MOVE #50 TO TEST NUMBER
1022 021500 004737 041460 JSR PC,CLDISK ;INIT AND SET UP GENERAL REG.
1023 021504 012777 000001 157546 MOV #DMD,@RHMR ;AND UNIT NUMBER
1024 ;SET DIAGNOSTIC MODE BIT
1025 ;THIS ENABLES COMMANDS WITHOUT MOL
1026 ;AND HOLDS RHLA FROM MOVING
1027
1028 021512 052777 000004 157540 ;GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
1029 021520 042777 000004 157532 BIS #MINX,@RHMR ;SET INDEX PULSE
1030 BIC #MINX,@RHMR ;CLEAR INDEX
1031
1032 021526 013777 001476 157504 MOV RETCL,@RHCS1 ;LOAD RETURN TO CENTER LINE COMMAND INTO RHCS1
1033
```

```

1034 ;SAVE REGISTERS FOR COMPARISON AFTER GO
1035 021534 004037 042152 JSR R0,SAVER ;SAVE
1036 021540 001232 RHCW ;FROM
1037 021542 002554 REINTC ;TO
1038 021544 000023 19. ;NUMBER OF REGISTERS SAVED
1039
1040 ;GIVE GO TO RETURN TO CENTER LINE COMMAND
1041 021546 052777 000001 157464 BIS #GO,@RHCS1 ;GO TO RETURN TO CENTER COMMAND
1042
1043
1044 ;FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CENTER LINE
1045 021554 012700 000004 MOV #4,R0 ;COUNTER
1046 021560 012777 000011 157472 5$: MOV #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
1047 021566 012777 000001 157464 MOV #DMD,@RHMR ;RESET SECTOR CLOCK
1048 021574 005300 DEC R0 ;COUNT
1049 021576 001370 BNE 5$ ;BRANCH IF NOT COMPLETE
1050
1051
1052 ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
1053 021600 052737 000001 002562 BIS #GO,REINTC+6 ;SAVED RHCS1
1054 021606 052737 020000 002604 BIS #PIP,REINTC+30 ;SAVED RHDS1
1055 021614 042737 000200 002604 BIC #DRY,REINTC+30 ;SAVED RHDS1
1056
1057 ;AFTER GO HAS BEEN GIVEN TO RETURN TO CENTER LINE COMMAND
1058 ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
1059 ;BE DONE
1060 021622 004037 042152 JSR R0,SAVER ;SAVE
1061 021626 001232 RHCW ;FROM
1062 021630 001510 WRFROM ;TO
1063 021632 000023 19. ;NUMBER OF REGISTERS SAVED
1064
1065 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1066 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1067 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1068 021634 113737 002601 001535 MOVB REINTC+25,WRFROM+25;SAVE UPPER RHAS
1069
1070
1071 ;COMPARE REGISTERS BEFORE RETURN TO CENTER LINE COMMAND
1072 ;WITH AFTER GO
1073 021642 004037 042354 JSR R0,COMPAR ;COMPARE
1074 021646 002554 REINTC ;GOOD BUFFER
1075 021650 001510 WRFROM ;TEST BUFFER
1076 021652 000023 19. ;NUMBER
1077 021654 021662 1$ ;RETURN FOR ERROR
1078 021656 021662 1$ ;SAME
1079 021660 021702 2$ ;RETURN FOR GOOD COMPARISON
1080
1081 021662 013705 046426 1$: MOV ERWORD,R5 ;GETTING READY TO INDEX
1082 021666 060505 ADD R5,R5 ;DOUBLE ERROR WORD
1083 021670 016537 001230 041130 MOV RHCW-2(R5),REGADR ;FAILING REGISTER ADDRESS
1084 021676 104001 EMT 1
1085 ;AFTER RETURN TO CENTER LINE COMMAND
1086 ;WITH GO IS GIVEN
1087 021700 000207 RTS PC ;RETURN TO COMPARISON
1088
1089 ;NOW GIVE INIT AND GET ALL GO AND PIP DOWN
1090

```



```

1091 021702 052712 000040      2$:   BIS      #CLR,@R2      ;RH INITILIZE
1092 021706 013712 001374      MOV      UNIT,@R2      ;REINSTATE UNIT NUMBER
1093 021712 012777 000001 157340  MOV      #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
1094                                     ;THIS ENABLES COMMANDS WITHOUT MOL
1095                                     ;AND HOLDS RHLA FROM MOVING
1096
1097                                     ;CHANGE REGISTERS TO EXPECTED VALUE
1098 021720 042737 000001 002562  BIC      #GO,REINTO+6  ;SAVED RHCS1
1099 021726 042737 020000 002604  BIC      #PIP,REINTO+30 ;SAVED RHDS1
1100 021734 052737 000200 002604  BIS      #DRY,REINTO+30 ;SAVED RHDS1
1101 021742 017737 157326 002616  MOV      @RHLA,REINTO+42;SAVED RHLA
1102
1103                                     ;AFTER INITIALIZE SAVE REGISTERS SO THAT
1104                                     ;COMPARES CAN BE DONE
1105 021750 004037 042152      JSR      R0,SAVER      ;SAVE
1106 021754 001232      RHWC      ;FROM
1107 021756 001510      WRFROM    ;TO
1108 021760 000023      19.      ;NUMBER OF REGISTERS SAVED
1109
1110                                     ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1111                                     ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1112                                     ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1113 021762 113737 002601 001535  MOVB     REINTO+25,WRFROM+25;SAVE UPPER RHAS
1114
1115
1116                                     ;COMPARE REGISTERS AFTER INITIALIZE
1117 021770 004037 042354      JSR      R0,COMPAR     ;COMPARE
1118 021774 002554      REINTO    ;GOOD BUFFER
1119 021776 001510      WRFROM    ;TEST BUFFER
1120 022000 000023      19.      ;NUMBER OF REGISTERS TO BE
1121                                     ;COMPARED
1122 022002 022010      3$      ;RETURN POINT FOR ERROR
1123 022004 022010      3$      ;SAME
1124 022006 022030      4$      ;RETURN POINT FOR GOOD COMPARISON
1125
1126
1127 022010 013705 046426      3$:   MOV      ERWORD,R5    ;GETTING READY TO INDEX
1128 022014 060505      ADD      R5,R5        ;DOUBLE ERROR WORD
1129 022016 016537 001230 041130  MOV      RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
1130 022024 104001      EMT      1
1131                                     ;CONTENTS AFTER GIVING AN
1132                                     ;INITIALIZE FOLLOWING A RETURN TO
1133                                     ;CENTER LINE COMMAND
1134 022026 000207      RTS      PC          ;RETURN TO COMPARISON
1135
1136 022030      4$:   ;GOOD COMPARISON
1137
1144
:*****
:*TEST 51      RECALIBRATE COMMAND TEST
:*THE RECALIBRATE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
:*THEN ALL REGISTERS WILL BE CHECKED
:*RH CLEAR WILL BE GIVEN
:*THEN ALL REGISTERS WILL BE CHECKED
:*****
TST51:  SCOPE
1145 022032 012706 001100      MOV      #STACK,SP    ;RESET STACK
1146 022036 012737 000051 001432  MOV      #51,TSTNM    ;MOVE #51 TO TEST NUMBER
    
```

```

1147 022044 004737 041460      JSR    PC,CLDISK      ;INIT AND SET UP GENERAL REG.
1148                                ;AND UNIT NUMBER
1149 022050 012777 000001 157202  MOV    #DMD,@RHMR    ;SET DIAGNOSTIC MODE BIT
1150                                ;THIS ENABLES COMMANDS WITHOUT MOL
1151                                ;AND HOLDS RHLA FROM MOVING
1152
1153
1154                                ;GIVE ONE INDEX PULSE TO CLEAR RHLA BEFORE THE START OF THIS TEST
1155
1156 022056 052777 000004 157174  BIS    #MINX,@RHMR    ;SET INDEX PULSE
1157 022064 042777 000004 157166  BIC    #MINX,@RHMR    ;CLEAR INDEX
1158
1159 022072 013777 001446 157140  MOV    RECALI,@RHCS1  ;LOAD RECALIBRATE COMMAND INTO RHCS1
1160
1161                                ;SAVE REGISTERS FOR COMPARISON AFTER GO
1162 022100 004037 042152      JSR    R0,SAVER      ;SAVE
1163 022104 001232                                ;FROM
1164 022106 002554                                ;TO
1165 022110 000023                                ;NUMBER OF REGISTERS SAVED
1166
1167                                ;GIVE GO TO RECALIBRATE COMMAND
1168 022112 052777 000001 157120  BIS    #GO,@RHCS1    ;GO TO RECALIBRATE COMMAND
1169
1170
1171                                ;FOUR SECTOR CLOCKS ARE GIVEN TO TAKE POSITIONER TO CYLINDER 0
1172 022120 012700 000004      MOV    #4,R0          ;COUNTER
1173 022124 012777 000011 157126  S$:  MOV    #MSTCK!DMD,@RHMR ;SET SECTOR CLOCK
1174 022132 012777 000001 157120  MOV    #DMD,@RHMR    ;RESET SECTOR CLOCK
1175 022140 005300                                DEC    R0              ;COUNT
1176 022142 001370                                BNE    $$              ;BRANCH IF NOT COMPLETE
1177
1178
1179                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
1180 022144 052737 000001 002562  BIS    #GO,REINTO+6   ;SAVED RHCS1
1181 022152 052737 020000 002604  BIS    #PIP,REINTO+30 ;SAVED RHDS1
1182 022160 042737 000200 002604  BIC    #DRY,REINTO+30 ;SAVED RHDS1
1183
1184                                ;AFTER GO HAS BEEN GIVEN TO RECALIBRATE COMMAND
1185                                ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
1186                                ;BE DONE
1187 022166 004037 042152      JSR    R0,SAVER      ;SAVE
1188 022172 001232                                ;FROM
1189 022174 001510                                ;TO
1190 022176 000023                                ;NUMBER OF REGISTERS SAVED
1191
1192                                ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1193                                ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1194                                ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1195 022200 113737 002601 001535  MOVB   REINTO+25,WRFROM+25;SAVE UPPER RHAS
1196
1197
1198                                ;COMPARE REGISTERS BEFORE RECALIBRATE COMMAND
1199                                ;WITH AFTER GO
1200 022206 004037 042354      JSR    R0,COMPAR     ;COMPARE
1201 022212 002554                                ;GOOD BUFFER
1202 022214 001510                                ;TEST BUFFER
1203 022216 000023                                ;NUMBER
  
```

```

1204 022220 022226      1$          ;RETURN FOR ERROR
1205 022222 022226      1$          ;SAME
1206 022224 022246      2$          ;RETURN FOR GOOD COMPARISON
1207
1208 022226 013705 046426 1$:  MOV    ERWORD,R5      ;GETTING READY TO INDEX
1209 022232 060505      ADD     R5,R5          ;DOUBLE ERROR WORD
1210 022234 016537 001230 041130 MOV    RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
1211 022242 104001      EMT     1
1212
1213
1214 022244 000207      RTS     PC              ;AFTER RECALIBRATE COMMAND
1215
1216
1217
1218 022246 052712 000040 2$:  BIS     #CLR,@R2      ;RH INITILIZE
1219 022252 013712 001374      MOV    UNIT,@R2        ;REINSTATE UNIT NUMBER
1220 022256 012777 000001 156774 MOV    #DMD,@RHMR      ;SET DIAGNOSTIC MODE BIT
1221
1222
1223
1224
1225 022264 042737 000001 002562 ;CHANGE REGISTERS TO EXPECTED VALUE
1226 022272 042737 020000 002604 BIC    #GO,REINTO+6    ;SAVED RHCS1
1227 022300 052737 000200 002604 BIC    #PIP,REINTO+30  ;SAVED RHDS1
1228 022306 017737 156762 002616 BIS    #DRY,REINTO+30  ;SAVED RHDS1
1229
1230
1231
1232 022314 004037 042152      MOV    @RHLA,REINTO+42;SAVED RHLA
1233 022320 001232      JSR    R0,SAVER        ;SAVE
1234 022322 001510      RHWC   ;FROM
1235 022324 000023      WRFROM ;TO
1236
1237
1238
1239
1240 022326 113737 002601 001535 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1241
1242
1243
1244 022334 004037 042354      ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1245 022340 002554      ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1246 022342 001510      MOV    REINTO+25,WRFROM+25;SAVE UPPER RHAS
1247 022344 000023      JSR    R0,COMPAR      ;COMPARE
1248
1249 022346 022354      REINTO ;GOOD BUFFER
1250 022350 022354      WRFROM ;TEST BUFFER
1251 022352 022374      19.    ;NUMBER OF REGISTERS TO BE
1252
1253 022354 013705 046426 3$:  MOV    ERWORD,R5      ;COMPARE
1254 022360 060505      ADD     R5,R5          ;GOOD BUFFER
1255 022362 016537 001230 041130 MOV    RHWC-2(R5),REGADR ;TEST BUFFER
1256 022370 104001      EMT     1              ;NUMBER OF REGISTERS TO BE
1257
1258
1259
1260 022372 000207      3$:  MOV    ERWORD,R5      ;COMPARE
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
  
```

```

1261
1262 022374          4$:                               ;GOOD COMPARISON
1263
1270
:*****
:*TEST 52          RELEASE COMMAND TEST
:*THE RELEASE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
:*THEN ALL REGISTERS WILL BE CHECKED
:*RH CLEAR WILL BE GIVEN
:*THEN ALL REGISTERS WILL BE CHECKED
:*****
TST52:  SCOPE
1271 022374 000004          MOV      #STACK,SP          ;RESET STACK
1272 022402 012706 001100  MOV      #52,TSTNM         ;MOVE #52 TO TEST NUMBER
1273 022410 004737 041460  JSR      PC,CLDISK         ;INIT AND SET UP GENERAL REG.
1274                                     ;AND UNIT NUMBER
1275 022414 012777 000001 156636  MOV      #DMD,@RHMR        ;SET DIAGNOSTIC MCDE BIT
1276                                     ;THIS ENABLES COMMANDS WITHOUT MOL
1277                                     ;AND HOLDS RHLA FROM MOVING
1278
1279 022422 013777 001452 156610  MOV      RELEASE,@RHCS1    ;LOAD RELEASE COMMAND INTO RHCS1
1280
1281          ;SAVE REGISTERS FOR COMPARISON AFTER GO
1282 022430 004037 042152          JSR      R0,SAVER          ;SAVE
1283 022434 001232          RHWC          ;FROM
1284 022436 002554          REINTO         ;TO
1285 022440 000023          19.          ;NUMBER OF REGISTERS SAVED
1286
1287          ;GIVE GO TO RELEASE COMMAND
1288 022442 052777 000001 156570  BIS      #GO,@RHCS1        ;GO TO RELEASE COMMAND
1289 022450 052777 000001 156602  BIS      #DMD,@RHMR        ;SET DMD TO HOLD RHLA
1290
1291          ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
1292          ;AFTER GO HAS BEEN GIVEN TO RELEASE COMMAND
1293 022456 052737 000001 002602  BIS      #DMD,REINTO+26;SAVED RHMR
1294 022464 017737 156604 002616  MOV      @RHLA,REINTO+42;SAVED RHLA
1295
1296
1297          ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
1298          ;BE DONE
1299 022472 004037 042152          JSR      R0,SAVER          ;SAVE
1300 022476 001232          RHWC          ;FROM
1301 022500 001510          WRFROM         ;TO
1302 022502 000023          19.          ;NUMBER OF REGISTERS SAVED
1303
1304          ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1305          ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1306          ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1307 022504 113737 002601 001535  MOVB    REINTO+25,WRFROM+25;SAVE UPPER RHAS
1308
1309
1310          ;COMPARE REGISTERS BEFORE RELEASE COMMAND
1311          ;WITH AFTER GO
1312 022512 004037 042354          JSR      R0,COMPAR        ;COMPARE
1313 022516 002554          REINTO         ;GOOD BUFFER
1314 022520 001510          WRFROM         ;TEST BUFFER
1315 022522 000023          19.          ;NUMBER
1316 022524 022532          1$          ;RETURN FOR ERROR
    
```

```
1317 022526 022532          1$          ;SAME
1318 022530 022552          2$          ;RETURN FOR GOOD COMPARISON
1319
1320 022532 013705 046426    1$:      MOV      ERWORD,R5      ;GETTING REAYD TO INDEX
1321 022536 060505          ADD      R5,R5          ;DOUBLE ERROR WORD
1322 022540 016537 001230 041130  MOV      RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS
1323 022546 104001          EMT      1
1324
1325
1326 022550 000207          RTS      PC              ;AFTER RELEASE COMMAND
1327 022552
1328
;*****
;*TEST 53      MAKE CURRENT CYLINDER = 0
;*****
TST53:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #53,TSTNM      ;MOVE #53 TO TEST NUMBER
        JSR      PC,CLDISK      ;INIT DRIVE
        MOV      #DMD,@RHMR      ;SET DIAGNOSTIC MODE
        JSR      R0,MAKECYL      ;DO SEEK COMMAND FOLLOWED BY AN INIT TO
        .WORD    0              ;CHANGE RHCC TO CYLINDER 0

;*****
;*TEST 54      LOOK AHEAD REGISTER
;*A SEARCH COMMAND IS GIVEN FOR CYLINDER 0, TRACK 0, SECTOR 21.
;*THE LOOK AHEAD REGISTER IS CHECKED AFTER INDEX PULSE
;*THE EXTENSION FIELD IS CHECKED IN EACH SECTOR AFTER
;*128 BYTES THEN AGAIN AFTER 128 MORE BYTES THEN AGAIN AFTER 256 MORE BYTES
;*THE SECTOR COUNT FIELD IS CHECKED AFTER EACH SECTOR
;*AT THE END ALL REGISTERS ARE CHECKED
;*****
TST54:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #54,TSTNM      ;MOVE #54 TO TEST NUMBER
        JSR      PC,CLDISK      ;INIT AND SET UP GENERAL REGISTERS

;THESE ARE REGULAR SET UPS FOR SEARCH COMMAND
        MOV      #21,@RHST      ;DESIRED SECTOR/TRACK REGISTER
        ;TRACK 0 SECTOR 21
        CLR      @RHCA          ;DESIRED CYLINDER =0
        MOV      #FMT22,@RHOF    ;FORMAT BIT=1 (16 BITS PER WORD)
        MOV      SERCH,@R1      ;FILL SEARCH COMMAND IN RHCS1

;NOW SAVE REGISTERS STARTING FROM RHWC IN WRITE FROM BUFFER
        JSR      R0,SAVER      ;SAVE REGISTERS FOR COMPARISON
        ;AT THE END OF THE SEARCH
        RHWC          ;START SAVING FROM RHWC
        REINTO      ;SAVE INTO REINTO
        19.          ;NUMBER OF REGISTERS SAVED

        JSR      PC,CHECKT      ;CHECK THAT DVA,RDY,DPR,DRY = 1
        TYPE      ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
        ;TINUE TESTING IF BOTH AREN'T TRUE
        HALT          ;STOP THE TEST

;NOW THE DIAGNOSTIC MODE BIT WILL BE SET
;AND THE SEARCH OPERATION STARTED
```

1329
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360

```

1361
1362 022676 005037 001200          CLR      STMP1          ;THIS WILL HAVE THE EXPECTED
1363                                     ;VALUE OF RHLA REGISTER
1364
1365 022702 013700 001260          MOV      RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADDR.
1366 022706 017703 156332          MOV      @RHDSST,R3 ;GET DESIRED SECTOR/TRACK REG.
1367 022712 042703 177400          BIC      #177400,R3 ;GET SECTOR ONLY
1368 022716 010337 052352          MOV      R3,SECTR   ;DUPLICATE SECTOR
1369 022722 012710 000001          MOV      #DMD,@R0   ;S
1370 022726 052777 000001 156304  BIS      #GO,@RHCS1 ;GO
1371 022734 052710 000010          BIS      #MSTCK,@R0 ;SET SECTOR CLOCK
1372 022740 042710 000010          BIC      #MSTCK,@R0 ;CLEAR SECTOR CLOCK
1373 022744 000240                    NOP                    ;ALLOW TIME BETWEEN SECTOR CLOCKS
1374 022746 052710 000010          BIS      #MSTCK,@R0 ;SET SECTOR CLOCK
1375 022752 042710 000010          BIC      #MSTCK,@R0 ;CLEAR SECTOR CLOCK
1376 022756 000240                    NOP                    ;ALLOW TIME BETWEEN SECTOR CLOCKS
1377 022760 052710 000014          BIS      #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
1378 022764 012710 000001          MOV      #DMD,@R0   ;RESET INDEX AND SECTOR CLOCK
1379 022770 005703                    TST      R3          ;IF SECTOR REQUIRED JUMP OUT
1380 022772 001555                    BEQ      11$         ;BRANCH OF SECTOR ZERO REQUIRED
1381
1382                                     ;AFTER THE INDEX PULSE RHLA WILL BE CHECKED TO BE ZERO
1383                                     ;AND STMP4 WILL BE SET UP TO COUNT BYTES
1384 022774 012737 001140 001206 1$:  MOV      #608.,$TMP4 ;THERE ARE 608 BYTES PER SECTOR
1385 023002 017737 156266 001126    MOV      @RHLA,$BDDAT ;SAVE RHLA
1386 023010 017737 156230 001320    MOV      @RHDSST,DST ;SAVE DESIRED SECTOR TRACK
1387 023016 023737 001200 001126    CMP      STMP1,$BDDAT ;RHLA SHOULD BE HAVE EXTENSION
1388                                     ;FIELD EQUAL TO ZERO
1389 023024 001414                    BEQ      2$         ;BRANCH IF GOOD
1390 023026 013737 052352 001202    MOV      SECTR,$TMP2 ;GET SECTOR SOUGHT
1391 023034 160337 001202                    SUB      R3,$TMP2   ;$TMP2 NOW HAS PRESENT SECTOR
1392 023040 012746 001140                    MOV      #608.,-(SP) ;NUMBER OF BYTES PER SECTOR
1393 023044 163716 001206                    SUB      $TMP4,(SP) ;-(SP)HAS PRESENT BYTE NUMBER
1394 023050 012637 001204                    MOV      (SP)+,$TMP3 ;PRESENT BYTE NUMBER
1395 023054 104024                    EMT      24
1396                                     ;SECTOR IS IN ERROR
1397
1398
1399                                     ;NOW THE 304 WORDS WILL START
1400                                     ;FOR FIRST BYTE CLOCK WILL BE INDEPENDENT OF
1401                                     ;SECTOR CLOCK THEN IT WILL COINCIDE FOREVER TILL
1402                                     ;THE BEGINNING OF NEXT SECTOR
1403
1404                                     ;ONE WORD ONLY THAT IS TWO BYTES
1405
1406 023056 012702 000010 2$:  MOV      #8.,R2      ;BYTE
1407 023062 012705 000002          MOV      #2,R5      ;BYTES PER WORD
1408 023066 000404                    BR      4$
1409 023070 052710 000012 3$:  BIS      #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
1410 023074 042710 000012          BIC      #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
1411 023100 052710 000002 4$:  BIS      #MCLK,@R0   ;SET CLOCK
1412 023104 042710 000002          BIC      #MCLK,@R0   ;CLEAR CLOCK
1413 023110 005302                    DEC      R2          ;BYTE COUNTER
1414 023112 001372                    BNE      4$         ;BRANCH IF BYTE NOT COMPLETE
1415 023114 005337 001206          DEC      $TMP4      ;BYTE COUNT DOWN
1416 023120 012702 000007          MOV      #7,R2      ;SETUP FOR SECOND BYTE
1417 023124 005305                    DEC      R5          ;IS WORD COMPLETE?
    
```

```

1418 023126 001360          BNE      3$          ;BRENCH IF NOT COMPLETE
1419                                     ;TO GIVE SECTOR CLOCK AND CLOCK
1420
1421                                     ;NOW 303 WORDS ARE LEFT ALL ARE IDENTICAL
1422                                     ;THAT IS 606 IDENTICAL BYTES WILL BE GIVEN
1423                                     ;RHLA WILL BE CHECKED STAR TO COUNT AFTER
1424                                     ;BEGINNING OF SECTOR PULSE
1425                                     ;AFTER 128 BYTES (2 BYTES ARE ALREADY GIVEN)
1426                                     ;SO 127 MORE
1427                                     ;THEN RHLA WILL BE CHECKED AFTER 128 MORE BYTES
1428                                     ;THEN RHLA WILL BE CHECKED AFTER 256 MORE BYTES
1429                                     ;THEN THE TOTAL OF 608 BYTES WILL BE COMPLETED
1430                                     ;AND RHLA WILL BE MADE READY FOR NEXT SECTOR
1431                                     ;AND RHLA WILL BE CHECKED
1432
1433 023130 012705 000100          MOV      #64.,R5          ;R5 WILL KEEP TRACK WHEN
1434                                     ;EXTENSION FIELD IS TO BE CHECKED
1435 023134 012701 000177          MOV      #127.,R1        ;FIRST TIME CHECK EXTENSION FIELD
1436                                     ;AFTER 127 MORE BYTES
1437 023140 012702 000007          5$:    MOV      #7,R2          ;CLOCKS PER BYTE COUNTER
1438 023144 052710 000012          BIS      #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
1439 023150 042710 000012          BIC      #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
1440 023154 052710 000002          6$:    BIS      #MCLK,@R0      ;SET CLOCK
1441 023160 042710 000002          BIC      #MCLK,@R0      ;RESET CLOCK
1442 023164 005302                                     DEC      R2              ;COUNT DOWN CLOCKS PER BYTE
1443 023166 001372                                     BNE      6$              ;BRANCH IF BYTE NOT COMPLETE
1444 023170 005337 001206          DEC      $TMP4           ;COUNT DOWN BYTES
1445 023174 001436                                     BEQ      10$             ;BRANCHOUT IF 608 BYTES DONE
1446 023176 005301                                     DEC      R1              ;COUNT DOWN NUMBER OF BYTES
1447                                     ;TO CHECK EXTENSION FIELD
1448 023200 001357          BNE      5$          ;BRANCH IF EXTENSION FIELD NOT
1449                                     ;TO BE CHECKED YET
1450
1451                                     ;NOW THE EXTENSION FIELD OF THE LOOK AHEAD REGISTER
1452                                     ;WILL BE CHECKED
1453
1454 023202 062737 000020 001200          ADD      #20,$TMP1        ;GET TO THE NEXT EXTENSION
1455 023210 017737 156060 001126          MOV      @RHLA,$BDDAT    ;GET RHLA FOR COMPARISON
1456
1457 023216 017737 156022 001320          MOV      @RHDST,DST      ;SAVE DESIRED SECTOR TRACK
1458 023224 023737 001200 001126          CMP      $TMP1,$BDDAT    ;CHECK VALUE OF RHLA
1459 023232 001414                                     BEQ      7$              ;BRANCH IF GOOD
1460 023234 013737 052352 001202          MOV      SECTR,$TMP2    ;GET SECTOR SOUGHT
1461 023242 160337 001202                                     SUB      R3,$TMP2        ;$TMP2 NOW HAS PRESENT SECTOR
1462 023246 012746 001140                                     MOV      #608.,-(SP)    ;NUMBER OF BYTES PER SECTOR
1463 023252 163716 001206                                     SUB      $TMP4,(SP)     ;(SP) HAS PRESENT BYTE NUMBER
1464 023256 012637 001204                                     MOV      (SP)+,$TMP3    ;PRESENT BYTE NUMBER
1465 023262 104025                                     EMT      25
1466                                     ;OF A SECTOR IS IN ERROR
1467
1468 023254 060505          7$:    ADD      R5,R5          ;GET NEXT STEP TO CHECK EXTENSION FIELD
1469 023266 010501          MOV      R5,R1          ;PUT IN COUNTER
1470 023270 000723          BR      5$              ;BRANCH BACK SECTOR
1471                                     ;IS NOT COMPLETE
1472 023272 062737 000020 001200          10$:   ADD      #20,$TMP1
1473 023300 052710 000010          BIS      #MSTCK,@R0      ;THESE TWO INSTRUCTIONS GIVE
1474 023304 042710 000010          BIC      #MSTCK,@R0      ;ONE SECTOR CLOCK EXTRA
    
```

```

1475 023310 000240          NOP          ;ALLOW TIME BETWEEN SECTOR CLOCK
1476 023312 052710 000010  BIS      #MSTCK,@R0 ;THESE TWO INSTRUCTIONS GIVE
1477 023316 042710 000010  BIC      #MSTCK,@R0 ;ONE SECTOR CLOCK EARLY
1478                                ;BEFORE THE NEXT SECTOR
1479 023322 005303          DEC      R3      ;IS REQUIRED NO OF SECTORS COMPLETE
1480 023324 001223          BNE      1$      ;BRANCH IF NOT
1481
1482                                ;NOW THE REQUIRED SECTOR IS REACHED
1483                                ;ONE SECTOR CLOCK WILL BE GIVEN TO GET SECTOR PULSE
1484                                ;DOWN AND HENCE ATA UP
1485
1486 023326 012702 000010  11$:    MOV      #8,R2      ;8 CLOCKS
1487 023332 052710 000002  12$:    BIS      #MCLK,@R0 ;SET CLOCK
1488 023336 042710 000002  BIC      #MCLK,@R0 ;CLEAR CLOCKS
1489 023342 005302          DEC      R2      ;COUND DOWN
1490 023344 001372          BNE      12$     ;BRANCH IF 8 NOT DONE
1491 023346 052710 000012  BIS      #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
1492 023352 042710 000012  BIC      #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
1493
1494                                ;NOW ALL REGISTERS WILL BE COMPARED
1495                                ;SO FILL EXPECTED VALUE INTO SAVED LOCATIONS
1496
1497 023356 052737 100000 002562  BIS      #SC,REINT0+6 ;INCLUDE SC IN SAVED RHCS1
1498 023364 053737 001416 002600  BIS      ATTENT,REINT0+24 ;FILL APPROPRIATE ATTENTION
1499                                ;IN SAVED RHAS
1500 023372 052737 000001 002602  BIS      #DMD,REINT0+26 ;SET DMD IN RHMR SAVED
1501 023400 052737 100000 002604  BIS      #ATA,REINT0+30 ;SET ATA IN RHDS1 SAVED
1502 023406 013737 001200 002616  MOV      $TMP1,REINT0+42 ;MOVE EXPECTED VALUE
1503                                ;INTO RHLA SAVED
1504
1505                                ;AFTER SEARCH COMMAND
1506                                ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
1507
1508 023414 004037 042152          JSR      R0,SAVER ;SAVE
1509 023420 001232          RHC      ;FROM
1510 023422 001510          WRFROM ;TO
1511 023424 000023          19. ;NUMBER
1512
1513                                ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1514                                ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1515                                ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1516 023426 113737 002601 001535  MOVB     REINT0+25,WRFROM+25;SAVE UPPER RHAS
1517
1518                                ;COMPARE REGISTERS BEFORE SEARCH WITH AFTER
1519
1520
1521 023434 004037 042354          JSR      R0,COMPAR ;COMPAR
1522 023440 002554          REINT0 ;GO BUFFER
1523
1524 023442 001510          WRFROM ;TEST BUFFER
1525 023444 000022          18. ;NUMBER
1526 023446 023454          13$ ;RETURN FOR ERROR
1527 023450 023454          13$ ;SAME
1528 023452 023474          14$ ;RETURN FOR GOOD COMPARISON
1529 023454 013705 046426  13$:    MOV      ERWORD,R5 ;GETTING READY TO INDEX
1530 023460 060505          ADD     R5,R5 ;DOUBLE ERROR WORD
1531 023462 016537 001230 041130  MOV      RHC-2(R5),REGADR ;FAILING REG. ADDRESS
    
```


1532 023470 104001
1533 023472 000207
1534 023474
1535

EMT 1
RTS PC :CHANGED AT END OF
14\$: :SEARCH
:*****
:*TEST 55 MAKE CURRENT CYLINDER = 0
:*****

023474 000004
023476 012706 001100
023502 012737 000055 001432
023510 004737 041460
023514 012777 000001 155536
023522 004037 044032
023526 000000

TST55: SCOPE
MOV #STACK,SP :RESET STACK
MOV #55,TSTNM :MOVE #55 TO TEST NUMBER
JSR PC,CLDISK :INIT DRIVE
MOV #DMD,@RHMR :SET DIAGNOSTIC MODE
JSR R0,MAKECYL :DO SEEK COMMAND FOLLOWED BY AN INIT TO
.WORD 0 :CHANGE RHCC TO CYLINDER 0

1
2
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

```
.SBTTL READ/WRITE ADDRESSING VIA RHMR
*****
*TEST 56 WRITE HEADER AND DATA 1
*WRITE CYLINDER 0, FORMAT 16 BIT PER WORD
*TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 0'S
*AS EVERYTHING IS ZERO THIS PROVES VERY LITTLE
*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST56: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #56,TSTNM ;MOVE #56 TO TEST NUMBER

MOV #SECGAP,R0 ;POINTER
MOV #304,R1 ;COUNTER
1$: MOV #-1,(R0)+ ;CLEAR 'DISK' AREA TO ALL ONES
DEC R1 ;
BNE 1$ ;
JSR PC,CLDISK ;THIS IS USED TO SET UP GENERAL
;REGISTER CORRESPONDENCE

;THESE ARE TO SET UP FOR DISKLESS USE ONLY
MOV #FMT22,WCYL ;FORMAT 22=16 BIT WORDS AND
;CYLINDER 0
CLR WSECTR ;TRACK=0, SECTOR=0
CLR WKEY1 ;KEY1=0
CLR WKEY2 ;KEY2=0
MOV #256, FNWORD ;256 DATAWORDS
JSR R5,CRC ;GO TO CALCULATE CRC
WCYL
GCRC

;THESE ARE REGULAR SETUPS FOR RH11 & 'WRFROM' OUTPUT BUFFER
MOV #-260,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;BUS ADDRESS TO BE
MOV R0,@RHBA ;BUFFER 'WRFROM'
MOV #259,R5 ;COUNTER
MOV #FMT22,(R0)+ ;FORMAT =16 BIT WORD

2$: CLR (R0)+ ;SECTOR=0, TRACK=0,KEYS=0, ALL DATA=0
DEC R5 ;& CYLINDER=0....SO CLEAR ALL 'WRFROM'
BNE 2$ ;CONTINUE IF ALL 259 NOT COMPLETE
CLR @RHDST ;TRACK=0, SECTOR=0

JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
;STOP THE TEST

HALT

MOV WRIFOR,@R1 ;GET READY FOR WRITE HEADER
;AND DATA WITH 62 IN RHCS1

CLR ERFLG$ ;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMAT BIT=1 16 BIT WORDS
```

023530 000004
 023532 012706 001100
 023536 012737 000056 001432
 023544 012700 050126
 023550 012701 000460
 023554 012720 177777
 023560 005301
 023562 001374
 023564 004737 041460
 023570 012737 010000 051444
 023576 005037 051446
 023602 005037 051450
 023606 005037 051452
 023612 012737 000400 051504
 023620 004537 042666
 023624 051444
 023626 051454
 023630 012777 177374 155374
 023636 012700 001510
 023642 010077 155366
 023646 012705 000403
 023652 012720 010000
 023656 005020
 023660 005305
 023662 001375
 023664 005077 155354
 023670 004737 041514
 023674 104401 062726
 023700 000000
 023702 013711 001464
 023706 005037 001406
 023712 012777 010000 155330

```
54 023720 005077 155326          CLR    @RHCA          ;CYLINDER 0
55 023724 004737 051270          JSR    PC,COMWHD      ;WRITE HEADER AND DATA FROM 'WRFROM'
56                                     ;INTO THE RMR REGISTER AND BACK INTO
57                                     ;CORE 'DISK' AREA
58
59 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR
60 ;PRINT OUTS FROM THE 'COMWHD' ROUTINE THAT MEANS
61 ;ALL HEADER ON DISK IS GOOD IE. ONLY DATA IS
62 ;TO BE CHECKED TO SEE IF IT IS ZERO
63
64 ;AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF THEY
65 ;ARE ALL ZEROS (ECC1 AND ECC2 MAY NOT BE 0) AS WELL AS
66 ;CHECKING RHC FOR ZERO
67
68 023730 017737 155276 001126     MOV    @RHWC,$BDDAT   ;MOVE WORD COUNTER INTO BAD DATA
69 023736 001401                    BEQ    5$             ;SHOULD HAVE COUNTED UP TO ZERO
70 023740 104040                    EMT    40
71                                     ;HEADER AND DATA
72
73 023742 005737 001406     5$:   TST    ERFLG$ ;HAVE ANY ERRORS OCCURED?
74 023746 001034                    BNE    TST57          ;:BRANCH IF YES
75 023750 004737 041704     JSR    PC,CHECKE     ;CHECK THAT BITS = 1
76 023754 104401 062726     TYPE  ,CPHALT       ;CANNOT CONTINUE TESTING IF THEY DON'T
77 023760 000000                    HALT                    ;STOP THE TEST
78 023762 005037 051224     CLR    WECC1         ;CLEAR ECC
79 023766 005037 051226     CLR    WECC2
80
81 ;REINTO BUFFER IS FILLED WITH EXPECTED DATA OF ALL 0'S
82
83 023772 004037 041376     JSR    R0,CLAREA     ;CLEAR 'REINTO'
84 023776 002554                    REINTO ;FROM
85 024000 003614                    REINTO+<272.*2> ;TO
86 024002 000000                    .WORD  0              ;FILL WITH ZEROS
87 024004 005037 001406     CLR    ERFLG$ ;CLEAR ERROR FLAG
88
89 ;COMPARE 'REINTO' BUFFER WITH 'DISK' BUFFER
90
91 024010 004037 042354     JSR    R0,COMPAR     ;CHECK
92 024014 002554                    REINTO ;GOOD BUFFER
93 024016 050224                    DISK   ;TEST BUFFER
94 024020 000421                    273.  ;NUMBER OF WORDS CHECKED
95 024022 024030                    3$    ;RETURN POINT FOR ERROR HEADER
96 024024 024034                    4$    ;RETURN POINT FOR ERROR DATA
97 024026 024040                    TST57 ;RETURN FOR GOOD COMPARISON
98 024030 104007                    3$:   EMT    7
99 024032 000207                    RTS    PC              ;RETURN TO 'COMPAR'
100 024034 104010                    4$:   EMT    10
101                                     ;DATA WORDS
102                                     ;257 AND 258 ARE ECC
103 024036 000207                    RTS    PC              ;ZEROED OUT
104                                     ;259 TO 273 TOLERANCE GAP
112
;*****
;*TEST 57 WRITE HEADER AND DATA 2
```

```

:*WRITE CYLINDER0, FORMAT 16 BITS PER WORD
:*TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
:*OF ALL ONES.
:*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*****
    
```

```

113 024040 000004
114 024042 012706 001100
115 024046 012737 000057 001432
116 024054 012700 050126
117 024060 012701 000460
118 024064 005020
119 024066 005301
120 024070 001375
121 024072 004737 041460
122
123
124
125 024076 012737 010000 051444
126
127 024104 012737 000001 051446
128 024112 005037 051450
129 024116 005037 051452
130 024122 012737 000400 051504
131 024130 004537 042666
132 024134 051444
133 024136 051454
134
135
136
137 024140 012777 177374 155064
138 024146 012700 001510
139 024152 010077 155056
140
141 024156 012720 010000
142
143 024162 012720 000001
144 024166 005020
145 024170 005020
146 024172 012705 000400
147 024176 012720 177777
148 024202 005305
149 024204 001374
150 024206 012777 000001 155030
151
152 024214 004737 041514
    024220 104401 062726
    024224 000000
153
154 024226 013711 001464
155
156 024232 005037 001406
157 024236 012777 010000 155004
158 024244 005077 155002
159 024250 004737 051270
    
```

```

TST57: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #57,TSTNM ;MOVE #57 TO TEST NUMBER

1$: MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
CLR (R0)+ ;CLEAR SIMULATED 'DISK' AREA IN CORE
DEC R1
BNE 1$
JSR PC,CLDISK ;THIS IS USED TO SET GENERAL REGISTERS

;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
MOV #FMT22,WCYL ;FORMAT 22 = 16 BIT WORDS AND
;CYLINDER 0
MOV #1,WSECTR ;TRACK=0, SECTOR=1
CLR WKEY1 ;KEY1=0
CLR WKEY2 ;KEY2=0
MOV #256.,FNWORD ;256 DATA WORDS
JSR R5,CRC ;GO TO CALCULATE CRC
WCYL
GCRC

;THESE ARE REGULAR SETUPS FOR THE RH11 AND 'WRFROM' BUFFER
MOV #-260.,@RHWC ;256 DATA WORDS 4 HEADER WORDS
MOV #WRFROM,R0 ;THESE TWO INSTRUCTIONS GETS
MOV R0,@RHBA ;ADDR. OF WRFROM INTO R0 AND
;BUS ADDRESS REGISTER
MOV #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
;CYLINDER=0
2$: MOV #1,(R0)+ ;TRACK=0, SECTOR=1, KEYS=0
CLR (R0)+ ;KEY1=0
CLR (R0)+ ;KEY2=0
MOV #256.,R5 ;COUNTER
3$: MOV #-1,(R0)+ ;MOVE ALL ONES FOR DATA
DEC R5
BNE 3$ ;BRANCH IF DATA NOT COMPLETE
MOV #1,@RHDST ;TRACK=0 SECTOR=1

JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
;STOP THE TEST

HALT

MOV WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
;DATA WITH 62 IN RHCS1
CLR ERFLG$ ;CLEAR ERROR FLAG
MOV #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
CLR @RHCA ;CYLINDER =0
JSR PC,COMWHD ;WRITE HEADER AND DATA INTO 'DISK' AREA
    
```

160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212

```

:IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
:FROM THE "COMWHD" ROUTINE THAT MEANS ALL THE HEADER ON "DISK"
:IS GOOD IE. ONLY THE DATA IS TO BE CHECKED TO SEE IF IT IS
:ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
:THEY ARE ALL ZEROS, - ECC1 AND ECC2 ARE NOT CHECKED.

:RHCW IS CHECKED TO BE = 0

MOV @RHCW,$BDDAT :LOAD WORD COUNTER JUST IN CASE
BEQ 6$ :SHOULD BE = 0
EMT 40

:HEADER AND DATA IS COMPLETED

6$: TST ERFLG$ ;HAVE ANY ERRORS OCCURRED?
BNE TST60 ;;BRANCH IF YES
JSR PC,CHECKE :CHECK THAT BITS = 1
TYPE ,CPHALT :CANNOT CONTINUE TESTING IF THEY DON'T
HALT :STOP THE TEST
CLR WECC1 :CLEAR ECC
CLR WECC2 :CLEAR ECC

:FILL 'REINTO' BUFFER WITH EXPECTED DATA OF ALL 1'S

JSR R0,CLAREA :FILL REINTO BUFFER
REINTO :FROM
REINTO+<255.*2> :TO
.WORD -1 :DATA
JSR R0,CLAREA :FILL REST
REINTO+<256.*2> :FROM
REINTO+<272.*2> :TO
0 :DATA

CLR ERFLG$ ;CLEAR ERROR FLAG

:NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER IN CORE

JSR R0,COMPAR :CHECK
REINTO :GOOD BUFFER
DISK :TEST BUFFER
273. :NUMBER OF WORDS CHECKED
4$ :RETURN POINT FOR ERROR HEADER
5$ :RETURN POINT FOR ERROR DATA
TST60 :RETURN FOR GOOD COMPARISON

4$: EMT 7
RTS PC :RETURN TO COMPARE

5$: EMT 10

:DATA WORDS
:WORD NOS 257 AND 258
:ARE ECC WHICH HAVE BEEN
:ZEROED
:WORD NOS 259
:IS DATA GAP
:WORD NOS 260 TO 273
:ARE TOLERANCE GAP
  
```

```

213 024374 000207          RTS      PC          ;RETURN TO COMPARE
214
222
*****
*TEST 60      WRITE HEADER AND DATA 3
*WRITE CYLINDER 0 FORMAT 16 BITS PER WORD
*TRACK 1, SECTOR 1, KEY 0, NUMBER OF WORDS 256
*ALTERNATE ONES AND ZEROS (052525)
*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
TST60:  SCOPE
223 024376 000004          MOV      #STACK,SP      ;RESET STACK
224 024400 012706 001100  MOV      #60,TSTNM      ;MOVE #60 TO TEST NUMBER
225 024404 012737 000060 001432
226 024412 012700 050126          MOV      #SECGAP,R0     ;POINTER
227 024416 012701 000460          MOV      #304.,R1      ;COUNTER
228 024422 012720 000377 1$:  MOV      #377,(R0)+    ;LOAD SIMULATED 'DISK' AREA WITH 377
229 024426 005301          DEC      R1             ;
230 024430 001374          BNE     1$             ;
231 024432 004737 041460          JSR     PC,CLDISK      ;THIS IS USED TO SET UP GENERAL
232                                     ;REGISTER CORRESPONDENCE
233
234                                     ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
235
236 024436 012737 010000 051444          MOV      #FMT22,WCYL   ;FORMAT 22=16 BIT WORDS AND
237                                     ;CYLINDER 0
238 024444 012737 000401 051446          MOV      #401,WSECTR   ;TRACK=1, SECTOR=1
239 024452 005037 051450          CLR     WKEY1          ;KEY1=0
240 024456 005037 051452          CLR     WKEY2          ;KEY2=0
241 024462 012737 000400 051504          MOV      #256.,FNWORD  ;256 DATA WORDS
242 024470 004537 042666          JSR     R5,CRC         ;GO TO CALCULATE CRC
243 024474 051444          WCYL
244 024476 051454          GCRC
245
246                                     ;THESE ARE REGULAR SETUPS FOR RH11 AND 'WRFROM' BUFFER
247
248 024500 012777 177374 154524          MOV      #-260.,@RHWC  ;256 DATA WORDS 4 HEADER WORDS
249 024506 012700 001510          MOV      #WRFROM,R0   ;THESE TWO INSTRUCTIONS GET
250                                     ;ADDR. OF WRFROM INTO R0
251 024512 010077 154516          MOV      R0,@RHBA     ;AND BUS ADDRESS REGISTER
252
253 024516 012720 010000          MOV      #FMT22,(R0)+ ;FORMAT=16 BIT WORDS
254                                     ;CYLINDER=0
255 024522 012720 000401 2$:  MOV      #401,(R0)+    ;TRACK=1, SECTOR=1, KEYS=0
256 024526 005020          CLR     (R0)+         ;KEY1=0
257 024530 005020          CLR     (R0)+         ;KEY2=0
258 024532 012705 000400          MOV      #256.,R5     ;COUNTER
259 024536 012720 052525 3$:  MOV      #052525,(R0)+ ;MOVE ALTERNATE ONES FOR DATA
260 024542 005305          DEC     R5            ;COUNT
261 024544 001374          BNE     3$           ;BRANCH IF DATA NOT COMPLETE
262 024546 012777 000401 154470          MOV      #401,@RHDST  ;TRACK=1 SECTOR=1
263
264 024554 004737 041514          JSR     PC,CHECKT     ;CHECK THAT DVA,RDY,DPR,DRY = 1
265 024560 104401 062726          TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
                                     ;TINUE TESTING IF BOTH AREN'T TRUE
265 024564 000000          HALT                 ;STOP THE TEST
    
```

```

266 024566 013711 001464      MOV    WRIFOR,@R1      ;GET READY FOR WRITE HEADER
267                                ;AND DATA WITH 62 IN RHCS1
268 024572 005037 001406      CLR    ERFLG$ ;CLEAR ERROR FLAG
269 024576 012777 010000 154444  MOV    #FMT22,@RHOF  ;FORMAT BIT=1(16 BIT WORDS
270 024604 005077 154442      CLR    @RHCA         ;CYLINDER=0
271 024610 004737 051270      JSR    PC,COMWHD     ;WRITE HEADER AND DATA INTO 'DISK' CORE
272
273                                ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
274                                ;FROM THE 'COMWHD' ROUTINE THAT MEANS ALL HEADER IN
275                                ;'DISK' IS GOOD. DATA IS TO BE CHECKED TO SEE
276                                ;IF IT IS ALL 052525 AND WRITE DATA GAP AND
277                                ;TOLERANCE GAP TO SEE IF THEY ARE ALL ZEROS.
278
279                                ;RHWC IS CHECKED TO BE = 0
280
281 024614 017737 154412 001126  MOV    @RHWC,$BDDAT  ;LOAD AND TEST FOR ZERO
282 024622 001401                      BEQ    6$            ;RHWC SHOULD = 0
283 024624 104040                      EMT    40
284
285                                ;HEADER AND DATA WAS COMPLETED
286
287                                ;ONLY ECC1 AND ECC2 ARE NOT CHECKED
288 024626 005737 001406 6$:   TST    ERFLG$ ;HAVE ANY ERRORS OCCURED?
289 024632 001041                      BNE    TST61        ;:BRANCH IF YES
290 024634 004737 041704          JSR    PC,CHECKE    ;CHECK THAT BITS = 1
      024640 104401 062726          TYPE   ,CPHALT     ;CANNOT CONTINUE TESTING IF THEY DON'T
      024644 000000                      HALT                    ;STOP THE TEST
291 024646 005037 051224          CLR    WECC1       ;CLEAR ECC
292 024652 005037 051226          CLR    WECC2       ;CLEAR ECC
293
294                                ;FILL 'REINTO' BUFFER WITH EXPECTED DATA
295
296 024656 004037 041376          JSR    R0,CLAREA   ;FILL REINTO BUFFER
297 024662 002554                      REINTO                ;FROM
298 024664 003552                      REINTO+<255.*2>      ;TO
      .WORD 52525                      ;DATA
299 024666 052525                      JSR    R0,CLAREA   ;FILL REST
300 024670 004037 041376          REINTO+<256.*2>    ;FROM
301 024674 003554                      REINTO+<272.*2>    ;TO
302 024676 003614                      .WORD 0             ;DATA
303 024700 000000                      CLR    ERFLG$ ;CLEAR ERROR FLAG
304 024702 005037 001406
305
306                                ;NOW COMPARE 'DISK' BUFFER WITH 'REINTO' BUFFER IN CORE
307
308 024706 004037 042354          JSR    R0,COMPAR   ;CHECK
309 024712 002554                      REINTO                ;GOOD BUFFER
310 024714 050224                      DISK                   ;TEST BUFFER
311 024716 000421                      273.                  ;NUMBER OF WORDS CHECKED
312 024720 024726                      4$                    ;RETURN POINT FOR ERROR HEADER
313 024722 024732                      5$                    ;RETURN PCINT FOR ERROR DATA
314 024724 024736                      TST61                 ;RETURN FOR GOOD COMPARISON
315 024726 104007 4$:   EMT    7
316 024730 000207                      RTS    PC             ;RETURN TO COMPARE
317 024732 104010 5$:   EMT    10
318                                ;DATA WORDS
    
```

```
319 ;WORD NOS 257 AND 258
320 ;ARE ECC WHICH HAVE BEEN
321 ;ZEROED
322 ;WORD NOS 259
323 ;IS DATA GAP
324 ;WORD NOS 260 TO 273
325 ;ARE TOLERANCE GAP
326 024734 000207 RTS PC ;RETURN TO COMPARE
327
334
*****
;*TEST 61 PROGRAM ERROR RHCS2 #10
;*WRITE CYLINDER 0, FORMAT 16 BIT PER WORD
;*TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 0'S
;*WHILE GO BIT IS SET ANOTHER GO IS GIVEN THIS SHOULD SET
;*PROGRAM ERROR
*****
TST61: SCOPE
335 024736 000004
336 024740 012706 001100 MOV #STACK,SP ;RESET STACK
337 024744 012737 000061 001432 MOV #61,TSTNM ;MOVE #61 TO TEST NUMBER
338 024752 012700 050126 MOV #SECGAP,R0 ;POINTER
339 024756 012701 000460 MOV #304,R1 ;COUNTER
340 024762 012720 177777 1$: MOV #-1,(R0)+ ;CLEAR DISK AREA TO ALL ONES.
341 024766 005301 DEC R1 ;
342 024770 001374 BNE 1$ ;
343 024772 004737 041460 JSR PC,CLDISK ;THIS IS USED TO SET GENERAL
344 ;REGISTERS
345
346 ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
347
348 024776 012737 010000 051444 MOV #FMT22,WCYL ;FORMAT 22=16 BITWORDS AND
349 ;CYLINDER 0
350 025004 005037 051446 CLR WSECTR ;TRACK=0, SECTOR=0
351 025010 005037 051450 CLR WKEY1 ;KEY1=0
352 025014 005037 051452 CLR WKEY2 ;KEY2=0
353 025020 012737 000400 051504 MOV #256, FNWORD ;256 DATAWORDS
354 025026 004537 042666 JSR R5,CRC ;GO TO CALCULATE CRC
355 025032 051444 WCYL
356 025034 051454 GCRC
357
358 ;THESE ARE REGULAR SETUPS
359
360 025036 012777 177374 154166 MOV #-260,@RHWC ;256 DATA WORDS 4 HEADER WORDS
361 025044 012700 001510 MOV #WRFROM,R0 ;FROM BUFFER 'WRFROM'
362 025050 010077 154160 MOV R0,@RHBA ;IN BUS ADDRESS
363 025054 012705 000403 MOV #259,R5 ;COUNTER
364 025060 012720 010000 MOV #FMT22,(R0)+ ;FORMAT =16 BIT WORD
365 ;CYLINDER=0
366 025064 005020 2$: CLR (R0)+ ;SECTOR=0, TRACK=0,KEYS=0, ALL DATA=0
367 025066 005305 DEC R5 ;COUNT
368 025070 001375 BNE 2$ ;BRANCH IF ALL 259 NOT COMPLETE
369 025072 005077 154146 CLR @RH DST ;TRACK=0, SECTOR=0
370
371 025076 004737 041514 JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
025102 104401 062726 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
025106 000000 HALT ;STOP THE TEST
```



```
372
373 025110 013711 001464      MOV    WRIFOR,@R1      ;GET READY FOR WRITE HEADER
374                                ;AND DATA WITH 62 IN RHCS1
375 025114 005037 001406      CLR    ERFLG$         ;CLEAR ERROR FLAG
376 025120 012777 010000 154122  MOV    #FMT22,@RHOF   ;FORMAT BIT=1 16 BIT WORDS
377 025126 005077 154120      CLR    @RHCA         ;CYLINDER 0
378 025132 012777 000001 154120  MOV    #DMD,@RHMR    ;SET DIAGNOSTIC MODE
379 025140 052777 000001 154072  BIS    #GO,@RHCS1    ;GO
380 025146 000240
381 025150 052777 000001 154062  BIS    #GO,@RHCS1    ;THIS GO SHOULD SET PGE
382
383 025156 004737 041064      JSR    PC,PUTREG      ;SAVE REGISTERS
384 025162 032737 002000 001314  BIT    #PGE,CS1      ;IS PGE SET
385 025170 001404
386 025172 013737 001236 001122  BEQ    3$             ;BRANCH IF GOOD
387 025200 104037      MOV    RHCS2,$BDADR
388                                EMT    37
389 025202                                ;WAS ATTEMPTED WITH ONE IN PROGRESS
3$:
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
29

```
.SBTTL MAINTAINANCE REGISTER TESTS THRU RHMR

:*****
:THE SECTOR GAP AND SYNC BYTE ARE ALWAYS READ AS
:ZEROS AND 144000 NO MATTER WHAT IS IN THE SIMULATED DISK AREA
:TAGGED SECGAP: AND WSSYNC:

:THE HEADER CONSISTING OF CYLINDER ADDRESS, SECTOR,
:TRACK AND THE KEYS ARE READ FROM LOCATION
:CYL:, SECTOR:, KEY1:, AND KEY2 AND NOT FROM
:HEADER: ON SIMULATED DISK

:CRC IS READ FROM SIMULATED DISK LOCATION WCRC:
:HEADER GAP IS ALWAYS READ AS ZEROS NO MATTER
:WHAT IS ON THE SIMULATED DISK AREA

:THE DATA SYNC IS READ FROM HDWSYN:
:ON SIMULATED DISK

:ALL DATA IS READ FROM SIMULATED DISK DISK:
:*****

:*****
:*TEST 62 READ HEADER AND DATA 1
:*READ CYLINDER 0 FORMAT 16 BITS PER WORD
:*TRACK 0, SECTOR 0, KEYS 0, 256 WORDS OF 0
:*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*****
```

30 025202 000004
 31 025204 012706 001100
 32 025210 012737 000062 001432
 33

```
TST62: SCOPE
        MOV #STACK,SP ;RESET STACK
        MOV #62,TSTNM ;MOVE #62 TO TEST NUMBER
```

```
:SETUP FOR WHAT IS TO BE READ
:HEADER CRC IS RESTORED FROM A SUBROUTINE

        MOV #0, -(SP) ;DATA TO BE READ
        MOV #256.,R5 ;COUNTER
        MOV #DISK,R0 ;START OF SIMULATED DISK DATA
1$:     MOV (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
        DEC R5 ;COUNT
        BNE 1$ ;BRANCH IF 256 NOT COMPLETE
        TST (SP)+ ;UNDO -(SP)
        MOV #17.,R5 ;2 ECC WORDS
        ;1 DATA GAP
        ;14 TOLERANCE GAP
2$:     CLR (R0)+ ;CLEAR ECC, DATA GAP, AND
        DEC R5 ;TOLERANCE GAP
        BNE 2$ ;BRANCH IF NOT COMPLETE
```

025216 012746 000000
 025222 012705 000400
 025226 012700 050224
 025232 011620
 025234 005305
 025236 001375
 025240 005726
 025242 012705 000021

025246 005020
 025250 005305
 025252 001375

```
:THESE ARE TO SETUP FOR DISKLESS USE ONLY

        MOV #0,FMT22,CYL ;16 BITS PER WORD
        ;CYLINDER 0, FORMAT 16 BITS
025254 012737 010000 046306
        MOV #0,SECOTR+1 ;TRACK 0
025262 112737 000000 046311
        MOV #0,SECOTR ;SECTOR 0
025270 112737 000000 046310
```

```

025276 012737 000000 046312      MOV      #0,KEY1 ;KEY1=0
025304 012737 000000 046314      MOV      #0,KEY2 ;KEY2=0
025312 012737 000400 046366      MOV      #256.,DAWORD ;NO. OF DATA WORDS
025320 005037 046316      CLR      X          ;THIS IS A READ COMMAND
025324 004537 042666      JSR      R5,CRC    ;GO TO CALCULATE CRC
025330 046306
025332 050206      CYL
                        WCRC
    
```

;THESE ARE REGULAR SETUPS FOR RH11 AND 'REINTO'

```

025334 004737 041460      JSR      PC,CLDISK ;SETUP GENERAL REGISTERS
025340 012777 177374 153664      MOV      #-256.-4.,@RHWC ;256. DATA & HEADER WORDS
025346 012777 002554 153660      MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
025354 112746 000000      MOV      #0,-(SP)  ;IN LOWER BYTE GET SECTOR
025360 112766 000000 000001      MOV      #0,1(SP) ;GET TRACK IN HIGHER BYTE
025366 012677 153652      MOV      (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
025372 012777 014000 153650      MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
                        ;ECC CORRECTION INHIBIT
                        ;BECAUSE ECC IS NOT GOING
                        ;TO BE CHECKED
                        ;CYLINDER 0

025400 005077 153646      CLR      @RHCA

025404 004737 041514      JSR      PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
025410 104401 062726      TYPE    ,CPHALT  ;AND THAT NO OTHERS = 1. CANNOT CON-
                        ;TINUE TESTING IF BOTH AREN'T TRUE
025414 000000      HALT          ;STOP THE TEST

025416 013711 001470      MOV      REFOR,@R1 ;READ HEADER AND DATA=72
025422 005037 001406      CLR      FRFLG$   ;CLEAR ERROR FLAG
025426 004737 046146      JSR      PC,COMHD ;READ HEADER AND DATA
    
```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
 ;DETECTED.

;RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION

```

025432 017737 153574 001126      MOV      @RHWC,$BDDAT ;LOAD AND TEST RHWC
025440 001401      BEQ     20$         ;SHOULD = 0
025442 104040      EMT     40
                        ;HEADER AND DATA
    
```

;HEADER AND DATA ARE TO BE CHECKED.
 ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 ;'WRFROM' IS FILLED WITH EXPECTED DATA AND
 ;COMPARISONS ARE MADE.

```

025444 005737 001406      20$:   TST      ERFLG$   ;ANY ERRORS ALREADY THERE
025450 001046      BNE     TST63      ;BRANCH IF YES
025452 004737 041704      JSR      PC,CHECKE ;CHECK THAT BITS = 1
025456 104401 062726      TYPE    ,CPHALT  ;CANNOT CONTINUE TESTING IF THEY DON'T
025462 000000      HALT          ;STOP THE TEST
025464 012700 001510      MOV      #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
025470 012720 010000      MOV      #0!FMT22,(R0)+ ;CYLINDER 0
    
```

```

025474 112746 000000      MOVB    #0,-(SP)      ;IN LOWER BYTE GET SECTOR
025500 112766 000000 000001  MOVB    #0,1(SP)     ;GET TRACK IN HIGHER BYTE
025506 012620          MOV     (SP)+,(R0)+  ;GET TRACK/SECTOR IN BUFFER
025510 012720 000000      MOV     #0,(R0)+    ;KEY1 IN BUFFER
025514 012720 000000      MOV     #0,(R0)+    ;KEY2 IN BUFFER
025520 012701 000400      MOV     #256.,R1    ;DATA WORD COUNTER
025524 012702 000000      MOV     #0,R2      ;DATA
025530 010220          3$: MOV     R2,(R0)+  ;DATA INTO BUFFER
025532 005301          DEC     R1          ;COUNT
025534 001375          BNE    3$          ;BRANCH IF 256 NOT DONE
    
```

:NOW READ DATA BUFFER WILL BE CHECKED

```

025536 004037 042354      JSR     R0,COMPAR   ;CHECK
025542 001510          WRFROM          ;GOOD BUFFER
025544 002554          REINTO         ;TEST BUFFER
025546 000404          4+256.       ;NUMBER OF WORDS CHECKED
025550 025556          4$          ;RETURN POINT FOR ERROR HEADER
025552 025562          5$          ;RETURN POINT FOR ERROR DATA
025554 025566          TST63        ;RETURN FOR GOOD COMPARISON
025556          4$: EMT     4
025556 104004          RTS     PC      ;RETURN TO 'COMPAR'
025560 000207          5$: EMT     5
025562          ;HEADER WORDS
025562 104005          ;5 TO 260 ARE DATA WORDS
025564 000207          RTS     PC      ;RETURN TO 'COMPAR'
    
```

34
41

```

:*****
:*TEST 63 READ HEADER AND DATA 2
:*READ CYLINDER 0 FORMAT 16 BITS PER WORD
:*TRACK 0, SECTOR 1, KEYS 0, 256 WORDS OF 177777
:*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*****
    
```

```

42 025566 000004          TST63: SCOPE
43 025570 012706 001100  MOV     #STACK,SP   ;RESET STACK
44 025574 012737 000063 001432  MOV     #63,TSTNM   ;MOVE #63 TO TEST NUMBER
45
    
```

:SETUP FOR WHAT IS TO BE READ
 :HEADER CRC IS RESTORED FROM A SUBROUTINE

```

025602 012746 177777      MOV     #-1,-(SP)   ;DATA TO BE READ
025606 012705 000400      MOV     #256.,R5   ;COUNTER
025612 012700 050224      MOV     #DISK,R0   ;START OF SIMULATED DISK DATA
025616 011620          1$: MOV     (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
025620 005305          DEC     R5        ;COUNT
025622 001375          BNE    1$        ;BRANCH IF 256 NOT COMPLETE
025624 005726          TST     (SP)+     ;UNDO -(SP)
025626 012705 000021      MOV     #17.,R5   ;2 ECC WORDS
                                ;1 DATA GAP
                                ;14 TOLERANCE GAP
025632 005020          2$: CLR     (R0)+  ;CLEAR ECC, DATA GAP, AND
025634 005305          DEC     R5        ;TOLERANCE GAP
    
```

```

025636 001375          BNE      2$          ;BRANCH IF NOT COMPLETE

;THESE ARE TO SETUP FOR DISKLESS USE ONLY

025640 012737 010000 046306      MOV      #0!FMT22,CYL      ;16 BITS PER WORD
                                ;CYLINDER 0, FORMAT 16 BITS
025646 112737 000000 046311      MOV      #0,SECOTR+1      ;TRACK 0
025654 112737 000001 046310      MOV      #1,SECOTR        ;SECTOR 1
025662 012737 000000 046312      MOV      #0,KEY1 ;KEY1=0
025670 012737 000000 046314      MOV      #0,KEY2 ;KEY2=0
025676 012737 000400 046366      MOV      #256.,DAWORD     ;NO. OF DATA WORDS
025704 005037 046316              CLR      X                ;THIS IS A READ COMMAND
025710 004537 042666              JSR      R5,CRC           ;GO TO CALCULATE CRC
025714 046306                    CYL
025716 050206                    WCRC

;THESE ARE REGULAR SETUPS FOR RH11 AND 'REINTO'

025720 004737 041460              JSR      PC,CLDISK        ;SETUP GENERAL REGISTERS
025724 012777 177374 153300      MOV      #-256.-4.,@RHWC  ;256. DATA 4 HEADER WORDS
025732 012777 002554 153274      MOV      #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
025740 112746 000001              MOV      #1,-(SP)        ;IN LOWER BYTE GET SECTOR
025744 112766 000000 000001      MOV      #0,1(SP)        ;GET TRACK IN HIGHER BYTE
025752 012677 153266              MOV      (SP)+,@RHDST     ;TRACK/SECTOR IN RHDST
025756 012777 014000 153264      MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
                                ;ECC CORRECTION INHIBIT
                                ;BECAUSE ECC IS NOT GOING
                                ;TO BE CHECKED
                                ;CYLINDER 0

025764 005077 153262              CLR      @RHCA

025770 004737 041514              JSR      PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
025774 104401 062726              TYPE     ,CPHALT        ;AND THAT NO OTHERS = 1. CANNOT CON-
                                ;TINUE TESTING IF BOTH AREN'T TRUE
026000 000000                    HALT                    ;STOP THE TEST

026002 013711 001470              MOV      REFOR,@R1       ;READ HEADER AND DATA=72
026006 005037 001406              CLR      ERFLG$         ;CLEAR ERROR FLAG
026012 004737 046146              JSR      PC,COMHD        ;READ HEADER AND DATA
    
```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 ;FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
 ;DETECTED.

;RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION

```

026016 017737 153210 001126      MOV      @RHWC,$BDDAT    ;LOAD AND TEST RHWC
026024 001401                    BEQ      20$             ;SHOULD = 0
026026 104040                    EMT      40
                                ;HEADER AND DATA
    
```

;HEADER AND DATA ARE TO BE CHECKED.
 ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 ;'WRFROM' IS FILLED WITH EXPECTED DATA AND
 ;COMPARISONS ARE MADE.

```

026030 005737 001406 20$: TST ERFLG$ ;ANY ERRORS ALREADY THERE
026034 001046 BNE TST64 ;BRANCH IF YES
026036 004737 041704 JSR PC,CHECKE ;CHECK THAT BITS = 1
026042 104401 062726 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
026046 000000 HALT ;STOP THE TEST
026050 012700 001510 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
026054 012720 010000 MOV #0!FMT22,(R0)+ ;CYLINDER 0
026060 112746 000001 MOV#B #1,-(SP) ;IN LOWER BYTE GET SECTOR
026064 112766 000000 000001 MOV#B #0,1(SP) ;GET TRACK IN HIGHER BYTE
026072 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
026074 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
026100 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
026104 012701 000400 MOV #256.,R1 ;DATA WORD COUNTER
026110 012702 177777 MOV #-1,R2 ;DATA
026114 010220 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
026116 005301 DEC R1 ;COUNT
026120 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
    
```

;NOW READ DATA BUFFER WILL BE CHECKED

```

026122 004037 042354 JSR R0,COMPAR ;CHECK
026126 001510 WRFROM ;GOOD BUFFER
026130 002554 REINTO ;TEST BUFFER
026132 000404 4+256. ;NUMBER OF WORDS CHECKED
026134 026142 4$ ;RETURN POINT FOR ERROR HEADER
026136 026146 5$ ;RETURN POINT FOR ERROR DATA
026140 026152 TST64 ;RETURN FOR GOOD COMPARISON
026142 4$: EMT 4
026142 104004 RTS PC ;RETURN TO 'COMPAR'
026144 000207 5$: EMT 5
026146 104005 ;HEADER WORDS
026150 000207 RTS PC ;5 TO 260 ARE DATA WORDS
;RETURN TO 'COMPAR'
    
```

46
53

```

*****
;*TEST 64 READ HEADER AND DATA 3
;*READ CYLINDER 0 FORMAT 16 BITS PER WORD
;*TRACK 1, SECTOR 1, KEYS 0, 256 WORDS OF 052525
;*ANY ERROR LOGIC INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
;*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
*****
    
```

```

54 026152 000004 TST64: SCOPE
55 026154 012706 001100 MOV #STACK,SP ;RESET STACK
56 026160 012737 000064 001432 MOV #64,TSTNM ;MOVE #64 TO TEST NUMBER
57
    
```

;SETUP FOR WHAT IS TO BE READ
 ;HEADER CRC IS RESTORED FROM A SUBROUTINE

```

026166 012746 052525 MOV #052525,-(SP) ;DATA TO BE READ
026172 012705 000400 MOV #256.,R5 ;COUNTER
026176 012700 050224 MOV #DISK,R0 ;START OF SIMULATED DISK DATA
026202 011620 1$: MOV (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
    
```

```

026204 005305          DEC      R5          :COUNT
026206 001375          BNE      1$          :BRANCH IF 256 NOT COMPLETE
026210 005726          TST      (SP)+       :UNDO -(SP)
026212 012705 000021  MOV      #17.,R5    :2 ECC WORDS
                                :1 DATA GAP
                                :14 TOLERANCE GAP
026216 005020          2$: CLR      (R0)+       :CLEAR ECC, DATA GAP, AND
026220 005305          DEC      R5          :TOLERANCE GAP
026222 001375          BNE      2$          :BRANCH IF NOT COMPLETE
    
```

:THESE ARE TO SETUP FOR DISKLESS USE ONLY

```

026224 012737 010000 046306  MOV      #0!FMT22,CYL :16 BITS PER WORD
                                :CYLINDER 0, FORMAT 16 BITS
026232 112737 000001 046311  MOVB     #1,SECOTR+1  :TRACK 1
026240 112737 000001 046310  MOVB     #1,SECOTR    :SECTOR 1
026246 012737 000000 046312  MOV      #0,KEY1 ;KEY1=0
026254 012737 000000 046314  MOV      #0,KEY2 ;KEY2=0
026262 012737 000400 046366  MOV      #256.,DAWORD :NO. OF DATA WORDS
026270 005037 046316          CLR      X            :THIS IS A READ COMMAND
026274 004537 042666          JSR      R5,CRC       :GO TO CALCULATE CRC
026300 046306          CYL
026302 050206          WCRC
    
```

:THESE ARE REGULAR SETUPS FOR RH11 AND 'REINTO'

```

026304 004737 041460          JSR      PC,CLDISK   :SETUP GENERAL REGISTERS
026310 012777 177374 152714  MOV      #-256.-4.,@RHWC :256. DATA 4 HEADER WORDS
026316 012777 002554 152710  MOV      #REINTO,@RHBA  :STARTING ADDRESS OF READ BUFFER
026324 112746 000001          MOVB     #1,-(SP)     :IN LOWER BYTE GET SECTOR
026330 112766 000001 000001  MOVB     #1,1(SP)    :GET TRACK IN HIGHER BYTE
026336 012677 152702          MOV      (SP)+,@RHDST :TRACK/SECTOR IN RHDST
026342 012777 014000 152700  MOV      #FMT22!ECI,@RHOF :16 BITS PER WORD
                                :ECC CORRECTION INHIBIT
                                :BECAUSE ECC IS NOT GOING
                                :TO BE CHECKED
                                :CYLINDER 0
026350 005077 152676          CLR      @RHCA
026354 004737 041514          JSR      PC,CHECKT   :CHECK THAT DVA,RDY,DPR,DRY = 1
026360 104401 062726          TYPE     ,CPHALT   :AND THAT NO OTHERS = 1. CANNOT CON-
                                :TINUE TESTING IF BOTH AREN'T TRUE
026364 000000          HALT      :STOP THE TEST
026366 013711 001470          MOV      REFOR,@R1   :READ HEADER AND DATA=72
026372 005037 001406          CLR      ERFLG$     :CLEAR ERROR FLAG
026376 004737 046146          JSR      PC,COMHD   :READ HEADER AND DATA
    
```

:IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 :FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
 :FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 :SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
 :DETECTED.

:RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION

```

026402 017737 152624 001126  MOV      @RHWC,$BDDAT :LOAD AND TEST RHWC
    
```

026410 001401 BEQ 20\$;SHOULD = 0
026412 104040 EMT 40 ;HEADER AND DATA

:HEADER AND DATA ARE TO BE CHECKED.
:IN CHECKING READ DATA THE WRITE FROM BUFFER
: 'WRFROM' IS FILLED WITH EXPECTED DATA AND
:COMPARISONS ARE MADE.

026414 005737 001406 20\$: TST ERFLG\$;ANY ERRORS ALREADY THERE
026420 001046 BNE TST65 ;BRANCH IF YES
026422 004737 041704 JSR PC,CHECKE ;CHECK THAT BITS = 1
026426 104401 062726 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF THEY DON'T
026432 000000 HALT ;STOP THE TEST
026434 012700 001510 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
026440 012720 010000 MOV #0!FMT22,(R0)+ ;CYLINDER 0
026444 112746 000001 MOV #1,-(SP) ;IN LOWER BYTE GET SECTOR
026450 112766 000001 000001 MOV #1,1(SP) ;GET TRACK IN HIGHER BYTE
026456 012620 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
026460 012720 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
026464 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
026470 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
026474 012702 052525 MOV #052525,R2 ;DATA
026500 010220 3\$: MOV R2,(R0)+ ;DATA INTO BUFFER
026502 005301 DEC R1 ;COUNT
026504 001375 BNE 3\$;BRANCH IF 256 NOT DONE

:NOW READ DATA BUFFER WILL BE CHECKED

026506 004037 042354 JSR R0,COMPAR ;CHECK
026512 001510 WRFROM ;GOOD BUFFER
026514 002554 REINTO ;TEST BUFFER
026516 000404 4+256. ;NUMBER OF WORDS CHECKED
026520 026526 4\$;RETURN POINT FOR ERROR HEADER
026522 026532 5\$;RETURN POINT FOR ERROR DATA
026524 026536 TST65 ;RETURN FOR GOOD COMPARISON
026526 104004 4\$: EMT 4 ;RETURN TO 'COMPAR'
026530 000207 RTS PC
026532 104005 5\$: EMT 5 ;HEADER WORDS
026534 000207 RTS PC ;5 TO 260 ARE DATA WORDS
;RETURN TO 'COMPAR''

58
65

:*TEST 65 WRITE DATA
:*WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
:*TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256 OF 377
:*ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED
:*****

66 026536 000004 TST65: SCOPE
67 026540 012737 000065 001432 MOV #65,TSTNM ;MOVE #65 TO TEST NUMBER
68 026546 012706 001100 MOV #STACK,SP ;RESET STACK


```

026552 012737 000065 001432      MOV      #65,TSTNM      ;MOVE #65 TO TEST NUMBER
026560 004037 041376              JSR      R0,CLAREA      ;CLEAR SIMLLATED DISK
026564 050224              .WORD   DISK            ;FROM
026566 051250              .WORD   TOLGAP+16      ;TO
026570 000000              .WORD   0              ;DATA

```

;THESE ARE SETUP FOR DISKLESS USE ONLY

```

026572 012737 010000 046306      MOV      #0!FMT22,CYL   ;CYLINDER 0
026600 112737 000000 046311      MOVB     #0,SECOTR+1    ;16 BITS PEP WORD
026606 112737 000000 046310      MOVB     #0,SECOTR      ;TRACK 0
026614 005037 046312              CLR      KEY1           ;SECTOR 0
026620 005037 046314              CLR      KEY2           ;KEY1 0
026624 012737 000400 046354      MOV      #256.,NOWORD   ;KEY2 0
026632 012737 000001 046316      MOV      #1,X           ;NO OF DATA WORDS
026640 004537 042666              JSR      R5,CRC         ;WRITE DATA
026644 046306              CYL                    ;GO TO CALCULATE CRC
026646 050206              WCRC

```

;THESE ARE REGULAR SETUPS

```

026650 004037 041376              JSR      R0,CLAREA      ;FILL WRITE BUFFER WITH 377
026654 001510              WRFROM                ;FROM LOCATION
026656 002510              WRFROM+<256.*2>       ;TO LOCATION
026660 000377              377                   ;DATA
026662 004737 041460              JSR      PC,CLDISK     ;SETUP GENERAL REGISTERS
026666 012777 177400 152336      MOV      #-256.,@RHWC   ;256. DATA WORDS
026674 012777 001510 152332      MOV      #WRFROM,@RHEA ;STARTING ADDRESS OF WRITE BUFFER
026702 012746 000000              MOV      #0,-(SP)      ;SECTOR 0
026706 112766 000000 000001      MOVB     #0,1(SP)      ;TRACK 0
026714 012677 152324              MOV      (SP)+,@RHDSY   ;SECTOR 0 TRACK 0
026720 012777 010000 152322      MOV      #FMT22,@RHOF   ;16 BITS PER WORD FORMAT
026726 012777 000000 152316      MOV      #0,@RHCA      ;CYLINDER 0
026734 004737 041514              JSR      PC,CHECKT     ;CHECK THAT DVA,RDY,DPR,DRY = 1
026740 104401 062726              TYPE      ,CPHALT     ;AND THAT NO OTHERS = 1. CANNOT CON-
                                ;TINUE TESTING IF BOTH AREN'T TRUE
026744 000000              HALT                  ;STOP THE TEST
026746 013711 001462              MOV      WRIDAT,@R1    ;WRITE DATA=60
026752 005037 001406              CLR      ERFLG$        ;CLEAR ERROR FLAG
026756 004737 046146              JSR      PC,COMHD      ;WRITE DATA

```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
;FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
;HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
;AND SYNC'S WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.

```

026762 004737 041064              JSR      PC,PUTREG     ;SAVE REGISTERS
026766 005737 001406              TST      ERFLG$        ;HAVE ANY ERRORS OCCURED?
026772 001041              BNE      TST66         ;BRANCH IF YES
026774 012700 000377              MOV      #377,R0       ;GOOD DATA
027000 012701 050224              MOV      #DISK,R1      ;DATA WRITTEN INTO 'DISK'
027004 012702 000400              MOV      #256.,R2      ;COUNTER
027010 012737 000401 046426 1$:   MOV      #256.+1,ERWORD ;FOR ERROR WORD
027016 020021              CMP      R0,(R1)+      ;COMPARE GOOD DATA WITH DATA ON DISK

```

```

027020 001424      BEQ      3$      ;BRANCH IF GOOD
027022 010037 001124  MOV      RO,$GDDAT ;GOOD DATA
027026 014137 001126  MOV      -(R1),$BDDAT ;BAD DATA
027032 160237 046426  SUB      R2,ERWORD  ;ERROR WORD NO
027036 005737 001406  TST      ERFLG$     ;ANY ERRORS ALREADY THERE?
027042 001002      BNE      2$      ;BRANCH IF YES
027044 104004      EMT      4
027046 000401      BR       1064$    ;BRANCH TO AVOID PRINTING NEXT ERROR
    
```

```

027050      2$:
027050 104005      EMT      5
027052 005721 1064$: TST      (R1)+    ;UNDO -(R1) FOR BAD DATA
027054 017746 152060  MOV      @SWR,-(SP) ;GET SWITCH SETTING
027060 042716 177177  BIC      #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
027064 022726 000200  CMP      #SW07,(SP)+ ;IS 7 SET AND 8 RESET
027070 001402      BEQ      TST66    ;BRANCH OUT IF YES
027072 005302      DEC      R2      ;IF NOT COUNT 256 WORDS
027074 001345      BNE      1$      ;BRANCH IF 256. NOT DONE
    
```

69
76

```

;*****
;*TEST 66      READ DATA
;*READ CYLINDER 0, FORMAT 16 BITS PER WORD
;*TRACK 0, SECTOR 1, KEYS 0, 10 WORDS OF 177400
;*ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE
;*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
;*****
    
```

```

77 027076 000004      TST66: SCOPE
027100 012706 001100  MOV      #STACK,SP ;RESET STACK
78
79 027104 012737 000066 001432  MOV      #66,TSTNM ;MOVE #66 TO TEST NUMBER
80 027112 004037 041376  JSR      RO,CLAREA ;CLEAR SIMULATED DISK
81 027116 050224      .WORD   DISK      ;FROM
82 027120 051222      .WORD   DISK+776 ;TO
83 027122 177400      .WORD   177400   ;DATA
84
85 027124 004037 041376  JSR      RO,CLAREA ;CLEAR READ INTO BUFFER
86 027130 002554      .WORD   REINTO    ;FROM
87 027132 003552      .WORD   REINTO+776 ;TO
88 027134 000000      .WORD   0        ;DATA
89
90      ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
91
92 027136 012737 010000 046306  MOV      #FMT22,CYL ;CYLINDER 0 16 BITS PER WORD FORMAT
93 027144 105037 046311      CLRB    SECOTR+1  ;TRACK 0
94 027150 112737 000001 046310  MOVB    #1,SECOTR ;SECTOR 1
95 027156 005037 046312      CLR     KEY1      ;KEY1=0
96 027162 005037 046314      CLR     KEY2      ;KEY2=0
97 027166 012737 000012 046366  MOV      #10.,DAWORD ;NO. OF DATA WORDS
98 027174 005037 046316      CLR     X        ;THIS IS A READ COMMAND
99 027200 004537 042666  JSR      R5,CRC ;GO TO CALCULATE CRC
100 027204 046306      CYL
101 027206 050206      WCRC
102
103      ;THESE ARE REGULAR SETUPS
104
105 027210 004737 041460      JSR      PC,CLDISK ;SETUP GENERAL REGISTERS
    
```



```

159                                     :ZERO IF OTHER THAN ZERO
160                                     :WRONG NUMBER OF WORDS HAVE
161                                     :BEEN READ IN THE DISK NOW
162                                     :CONTAINS 177400 ALL 256
163                                     :WORDS BUT ONLY 10 WORDS
164                                     :SHOULD BE READ IN
165
166 027416 022021      4$:  CMP      (R0)+,(R1)+  :UNDO -(R0) AND -(R1) FOR ERROR
167 027420 017746 151514  MOV      @SWR,-(SP)  :GET SWITCH SETTING
168 027424 042716 177177  BIC      #177177,(SP) :KEEP ONLY SWITCH 7 AND 8
169 027430 022726 000200  CMP      #SW07,(SP)+ :IS 7 SET AND 8 RESET
170 027434 001402      BEQ      TST67      :BRANCH OUT IF YES
171 027436 005302      2$:  DEC      R2          :COUNT
172 027440 001345      BNE      1$          :BRANCH IF NOT COMPLETE
173
181
  
```

```

:*****
:*TEST 67 WRITE CHECK HEADER AND DATA
:*WRITE CHECK CYLINDER 0, FORMAT 16 BITS PER WORD
:*TRACK 1, SECTOR 1, KEYS 0, 36 WORDS AS SHOWN BELOW
:*ANY DEVICE LOGIC ERROR INDICATION IS NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*ONLY RH WRITE CHECK ERROR (RHCS2 BIT 14) IS TESTED HERE
:*****
TST67: SCOPE
  
```

```

182 027442 000004
183
184 :DATA TABLE
185 :20 WORDS OF 070707
186 :THEN 16 WORDS WITH ZERO FLOATING FROM RIGHT
187 :TO LEFT (EG. 177776,177775,177773 ETC)
188 027444 012706 001100      MOV      #STACK,SP  :RESET STACK
189 027450 012737 000067 001432  MOV      #67,TSTNM  :MOVE #67 TO TEST NUMBER
190
191 :SET UP 'REINTO' FOR WHAT IS TO BE READ
192 027456 012701 002554      MOV      #REINTO,R1  :STARTING ADDRESS
193 027462 012721 010000      MOV      #FMT22,(R1)+ :CYLINDER 0 FORMAT 16 BIT WORDS
194 027466 012721 000401      MOV      #401,(R1)+  :TRACK=1, SECTOR=1
195 027472 005021      CLR      (R1)+      :KEY1=0
196 027474 005021      CLR      (R1)+      :KEY2=0
197 027476 004037 041,76      JSR      R0,CLAREA   :FILL 'REINTO' BUFFER
198 027502 002564      .WORD   REINTO+<4*2> :FROM
199 027504 002632      .WORD   REINTO+<23.*2> :TO
200 027506 070707      .WORD   070707      :DATA
201
202 027510 012700 177776      MOV      #177776,R0  :GETTING READY TO FLOAT 0
203 027514 012701 002634      MOV      #REINTO+<24.*2>,R1 :STARTING ADDRESS WHERE 177776 GOES
204 027520 010021      1$:  MOV      R0,(R1)+   :MOVE IN FLOATING 0
205 027522 000261      SEC      :SET CARRY
206 027524 006100      ROL      R0          :GET 0 ONE BIT LEFT
207 027526 103774      BCS     1$          :BRANCH IF 16 NOT DONE
208
209 027530 004037 041376      JSR      R0,CLAREA   :FILL THE REST OF BUFFER WITH 0
210 027534 002674      .WORD   REINTO+<40.*2> :FROM
211 027536 003552      .WORD   REINTO+776    :TO
212 027540 000000      .WORD   0            :DATA
213
214 :SET UP SIMULATED DISK WITH WHAT IS TO BE READ
  
```

```

215
216 027542 004037 041376      JSR    R0,CLAREA      ;FILL 'DISK' BUFFER
217 027546 050224              .WORD  DISK           ;FROM
218 027550 050272              .WORD  DISK+<19.*2>  ;TO
219 027552 070707              .WORD  070707        ;DATA
220
221 027554 012700 177776      MOV    #177776,R0     ;GETTING READY TO FLOAT ZEROS
222 027560 012701 050274      MOV    #DISK+<20.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
223 027564 010021      2$:  MOV    R0,(R1)+      ;MOVE IN FLOATING 0
224 027566 000261              SEC                   ;SET CARRY
225 027570 006100              ROL    R0             ;GET 0 ONE BIT LEFT
226 027572 103774              BCS    2$            ;BRANCH IF 16 NOT DONE
227
228 027574 004037 041376      JSR    R0,CLAREA      ;FILL THE REST OF BUFFER WITH 177777
229 027600 050334              .WORD  DISK+<36.*2>  ;FROM
230 027602 051222              .WORD  DISK+776      ;TO
231 027604 177777              .WORD  177777        ;DATA
232
233 027606 004737 042204      JSR    PC,WRCHHD      ;WRITE CHECK HEADER AND DATA
234                          ;CYLINDER 0, TRACK 1, SECTOR 1
235
236                          ;IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
237                          ;HAS BEEN COMPLETED NOW WRITE CHECK ERRGR BIT IS TO BE TESTED
238
239 027612 013746 001374      MOV    UNIT,-(SP)     ;GET UNIT NUMBER
240 027616 052716 000100      BIS    #IR,(SP)      ;ONLY BIT 6 SHOULD BE SET
241 027622 004737 041064      JSR    PC,PUTREG      ;SAVE REGISTERS
242 027626 022637 001312      CMP    (SP)+,CS2     ;COMPARE RHCS2
243 027632 001406              BEQ    4$            ;BRANCH IF GOOD
244 027634 032712 04C000      BIT    #WCE,@R2      ;WRITE CHECK ERROR HIGH?
245 027640 001402              BEQ    3$            ;BRANCH IF ERROR NOT DUE TO 'WCE'
246 027642 104017              EMT    17
247 027644 000401              BR     4$            ;RHBA CONTAINS ADDRESS+2
248                          ;OF THE WORD IN MEMORY FROM
249                          ;THE DISK THAT DID NOT COMPARE
250                          ;TRE AND SC WILL BE SET DUE TO
251                          ;WCE
252 027646 104017      3$:  EMT    17
253                          ;BITS OTHER THAN IR
254                          ;AND UNIT NO. WAS SET
255
256                          ;NOW CHECK MEMORY TO SEE IF NOTHING GOT DESTROYED
257                          ;FILL 'WRFROM' WITH WHAT SHOULD BE IN 'REINTO' THEN CHECK
258
259 027650 012700 001510      4$:  MOV    #WRFROM,R0    ;STARTING ADDRESS
260 027654 012720 010000      MOV    #FMT22,(R0)+  ;CYLINDER
261 027660 012720 000401      MOV    #401,(R0)+    ;TRACK=1, SECTOR=1
262 027664 005020              CLR    (R0)+         ;KEY1=0
263 027666 005020              CLR    (R0)+         ;KEY2=0
264
265 027670 004037 041376      JSR    R0,CLAREA      ;FILL 'WRFROM' BUFFER
266 027674 001520              .WORD  WRFROM+<4*2>  ;FROM
267 027676 001566              .WORD  WRFROM+<23.*2> ;TO
268 027700 070707              .WORD  070707        ;DATA
269
270 027702 012700 177776      MOV    #177776,R0     ;GETTING READY TO FLOAT 0
    
```

```
271 027706 012701 001570      MOV    #WRFROM+<24.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
272 027712 010021      5$:  MOV    RO,(R1)+      ;MOVE IN FLOATING 0
273 027714 000261      SEC                    ;SET CARRY
274 027716 006100      ROL    RO              ;GET 0 ONE BIT LEFT
275 027720 103774      BCS    5$             ;BRANCH IF 16 NOT DONE
276
277 027722 004037 041376      JSR    RO,CLAREA      ;FILL THE REST OF BUFFER WITH 0
278 027726 001630      .WORD WRFROM+<40.*2> ;FROM
279 027730 002506      .WORD WRFROM+776     ;TO
280 027732 000000      .WORD 0              ;DATA
281
282      ;NOW THE READ BUFFER WILL BE CHECKED
283 027734 005037 001406      CLR    ERFLG$        ;CLEAR ERROR FLAG
284
285 027740 004037 042354      JSR    RO,COMPAR     ;CHECK
286 027744 001510      WRFROM                ;GOOD BUFFER
287 027746 002554      REINTO                ;TEST BUFFER
288 027750 000400      256.                 ;NUMBER OF WORDS CHECKED
289 027752 027760      6$                   ;RETURN POINT FOR ERROR HEADER
290 027754 027764      7$                   ;RETURN POINT FOR ERROR DATA
291 027756 027772      TST70                 ;RETURN FOR GOOD COMPARISON
292 027760
293 027762 104004 000207      6$:  EMT    4
294 027764 027764 104005      RTS    PC             ;RETURN TO COMPARISON SUBROUTINE
295
296      ;CHANGED AFTER A WRITE
297      ;CHECK HEADER AND DATA COMMAND
298      ;WORD NO CONTAINS THE WORD
299      ;NUMBER THAT GOT CHANGED
300      ;RETURN TO COMPARISON SUBROUTINE
301 027770 000240      7$:  RTS    PC
302
303      ;ONLY A BRANCH POINT
304
305      ;*****
306      ;*TEST 70      WRITE CHECK DATA
307      ;*WRITE CHECK DATA CYLINDER 0 FORMAT 16 BITS PER WORD
308      ;*TRACK 1, SECTOR 1, KEYS 0, 32 WORDS OF DATA
309      ;*ANY DEVICE LOGIC ERROR INDICATIONS ARE NOT CONCLUSIVE ON FIRST PASS
310      ;*BECAUSE ERROR LOGIC HAS NOT YET (ON FIRST PASS) BEEN CHECKED
311      ;*ONLY RH WRITE CHECK ERROR IS TESTED
312      ;*****
313      027772 000004      TST70: SCOPE
314
315      ;DATA TABLE
316      ;TOTAL OF 32 WORDS CONSISTING OF
317      ;16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)
318      ;16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)
319
320      MOV    #STACK,SP    ;RESET STACK
321      MOV    #70,TSINM   ;MOVE #70 TO TEST NUMBER
322
323      ;SET LP 'REINTO' FOR WHAT IS TO BE READ
324
325      MOV    #1,RO        ;GETTING READY TO FLOAT 1
326      MOV    #REINTO,R1  ;STARTING ADDRESS WHERE 1 GOES
327      1$:  MOV    RO,(R1)+ ;MOVE FLOATING 1
```

```

325 030020 006100          ROL    RO          ;GET 1 ONE BIT LEFT
326 030022 103375          BCC    1$          ;BRANCH IF 16 NOT DONE
327 030024 012700 177776  MOV    #177776,RO  ;GETTING READY TO FLOAT 0
328 030030 012701 002614  MOV    #REINTO+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
329 030034 010021          MOV    RO,(R1)+    ;MOVE IN FLOATING 0
330 030036 000261          SEC          ;SET CARRY
331 030040 006100          ROL    RO          ;GET 0 ONE BIT LEFT
332 030042 103774          BCS    2$          ;BRANCH IF 16 NOT DONE
333
334 030044 004037 041376  JSR    RO,CLAREA   ;FILL REST OF BUFFER WITH 1
335 030050 002654          .WORD  REINTO+<32.*2> ;FROM
336 030052 003552          .WORD  REINTO+776    ;TO
337 030054 000001          .WORD  1            ;WITH DATA
338
339          ;SET UP SIMULATED DISK WITH WHAT IS TO BE READ
340
341 030056 012700 000001  MOV    #1,RO       ;GETTING READY TO FLOAT 1
342 030062 012701 050224  MOV    #DISK,R1    ;STARTING ADDRESS WHERE 1 GOES
343 030066 010021          MOV    RO,(R1)+    ;MOVE FLOATING 1
344 030070 006100          ROL    RO          ;GET 1 ONE BIT LEFT
345 030072 103375          BCC    3$          ;BRANCH IF 16 NOT DONE
346
347 030074 012700 177776  MOV    #177776,RO  ;GETTING READY TO FLOAT 0
348 030100 012701 050264  MOV    #DISK+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
349 030104 010021          MOV    RO,(R1)+    ;MOVE FLOATING 0
350 030106 000261          SEC          ;SET CARRY
351 030110 006100          ROL    RO          ;GET 0 ONE BIT LEFT
352 030112 103774          BCS    4$          ;BRANCH IF 16 NOT DONE
353
354 030114 004037 041376  JSR    RO,CLAREA   ;FILL REST OF BUFFER WITH 0
355 030120 050324          .WORD  DISK+<32.*2> ;FROM
356 030122 051222          .WORD  DISK+776    ;TO
357 030124 000000          .WORD  0            ;WITH DATA
358
359 030126 004737 042516  JSR    PC,WRCHDA   ;WRITE CHECK DATA
360          ;CYLINDER 0, TRACK 1, SECTOR 1
361          ;KEYS 0, 32 WORDS.
362
363          ;IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
364          ;HAS BEEN COMPLETED NOW WRITE CHECK ERROR BIT IS TESTED
365
366 030132 013746 001374  MOV    UNIT,-(SP)  ;GET UNIT NUMBER
367 030136 052716 000100  BIS    #IR,(SP)    ;ONLY BIT 6 SHOULD BE SET
368 030142 004737 041064  JSR    PC,PUTREG   ;SAVE REGISTERS
369 030146 022637 001312  CMP    (SP)+,CS2   ;COMPARE RHCS2
370 030152 001407          BEQ    6$          ;BRANCH IF GOOD
371 030154 032737 040000 001312  BIT    #WCE,CS2    ;WRITE CHECK ERROR HIGH?
372 030162 001402          BEQ    5$          ;BRANCH IF ERROR NOT DUE TO 'WCE'
373 030164 104017          EMT    17
374 030166 000401          BR     6$
375          ;RHBA CONTAINS ADDRESS+2
376          ;OF THE WORD IN MEMORY FROM
377          ;THE DISK THAT DID NOT COMPARE
378 030170          5$:          ;TRE AND SC WILL BE SET DUE TO WCE
379 030170 104017          EMT    17
380          ;BUT SOME BITS OTHER THAN
          ;IR AND UNIT NO. WERE SET

```

```

381
382
383           ;NOW CHECK MEMORY TO SEE IF ANYTHING GOT DESTROYED
384           ;FILL 'WRFROM' WITH WHAT SHOULD BE IN REINTO THEN CHECK IT
385 030172 005037 001406 6$: CLR ERFLG$ ;CLEAR ERROR FLAG
386 030176 012700 000001 MOV #1,R0 ;GETTING READY TO FLOAT 1
387 030202 012701 001510 MOV #WRFROM,R1 ;START ADDRESS WHERE 1 GOES
388 030206 010021 7$: MOV R0,(R1)+ ;MOVE FLOATING 1
389 030210 006100 ROL R0 ;GET 1 ONE BIT LEFT
390 030212 103375 BCC 7$ ;BRANCH IF 16 NOT DONE
391
392 030214 012700 177776 MOV #177776,R0 ;GETTING READY TO FLOAT 0
393 030220 012701 001550 MOV #WRFROM+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
394 030224 010021 10$: MOV R0,(R1)+ ;MOVE IN FLOATING 0
395 030226 000261 SEC ;SET CARRY
396 030230 006100 ROL R0 ;GET 0 ONE BIT LEFT
397 030232 103774 BCS 10$ ;BRANCH IF CARRY SET
398
399 030234 004037 041376 JSR R0,CLAREA ;FILL REST OF BUFFER WITH 1
400 030240 001610 .WORD WRFROM+<32.*2> ;FROM
401 030242 002506 .WORD WRFROM+776 ;TO
402 030244 000001 .WORD 1 ;WITH DATA
403
404           ;NOW THE READ BUFFER WILL BE CHECKED
405
406 030246 004037 042354 JSR R0,COMPAR ;CHECK
407 030252 001510 WRFROM ;GOOD BUFFER
408 030254 002554 REINTO ;TEST BUFFER
409 030256 000400 256. ;NUMBER OF WORDS CHECKED
410 030260 030266 11$ ;RETURN POINT FOR ERROR HEADER
411 030262 030272 12$ ;RETURN POINT FOR ERROR DATA
412
413 030264 030300 TST71 ;RETURN FOR GOOD COMPARISON
414
415 030266 11$: EMT 4
416 030270 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
417 030272 12$: EMT 5
418           ;CHANGED AFTER A WRITE
419           ;CHECK DATA COMMAND
420           ;WORD NO CONTAINS THE WORD
421           ;NUMBER THAT GOT CHANGED
422 030274 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
423
424 030276 000240 13$: NOP ;ONLY A BRANCH POINT

```


1
2
14

.SBTTL ERROR BIT FUNCTIONAL TESTS

 *TEST 71 ATTENTION WITH ERROR TEST
 *THIS TESTS THE SETTING OF ATA BIT BOTH IN THE RHAS
 *AND THE RHDS1 REGISTERS WITH THE SETTING OF EACH
 *ERROR BIT ON THE THREE ERROR REGISTERS.
 *IN EACH OF THE ABOVE CASES ERR IN RHDS1 SHOULD
 *ALSO SET.

*'GO' SHOULD CLEAR ERR, ATA IN RHDS1 AND RHAS BUT NOT ERROR REG.
 *PUTTING '1' IN RHAS DRIVE POSITION CLEARS DRIVE BIT IN ATA IN RHDS1
 *UPPER BYTE OF RHAS IS INVALID

```

15 030300 000004
16 030302 012706 001100
17 030306 012737 000071 001432
18 030314 004737 041460
    030320 004737 041514
    030324 104401 062726
19 030330 000000
20
21
22 030332 012700 002554
23
24 030336 013720 001242
25 030342 012720 000000
26 030346 013720 001246
27 030352 012720 000000
28 030356 013720 001254
29 030362 012720 000000
30
31 030366 013704 001256
32 030372 013705 001416
33 030376 012737 030424 001110
34
35 030404 012737 000003 001200
36 030412 012700 002554
37
38 030416 012002
39 030420 012701 000001 1$:
40 030424 052777 000040 150604 2$:
41 030432 013777 001374 150576
42 030440 010112
43 030442 004737 041064
44 030446 120537 001332
45
46 030452 001401
47 030454 104020
48
49
50
51
52 030456 013746 001336 3$:
53 030462 042716 001100
    
```

```

TST71: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #71,TSTNM ;MOVE #71 TO TEST NUMBER
JSR PC,CLDISK ;CLEAR DISK REGISTERS
JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
HALT ;STOP THE TEST

MOV #REINTO,R0 ;BUFFER STARTING FOR 3 ERROR
;REGISTERS
MOV RHER1,(R0)+ ;RHER1 STORED IN REINTO
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER1
MOV RHER2,(R0)+ ;RHER2 STORED IN REINTO+4
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER2
MOV RHER3,(R0)+ ;RHER3 STORED IN REINTO+10
MOV #0,(R0)+ ;BITS NOT TO BE CHECKED IN RHER3

MOV RHAS,R4 ;R4 HAS RHAS
MOV ATTENT,R5 ;R5 HAS ATA BIT IN RHAS
MOV #2$, $LPERR ;THAT SHOULD SET WITH ERROR
;RETURN POINT TO ERROR
MOV #3,$TMP1 ;ERROR REGISTER COUNTER
MOV #REINTO,R0 ;REGISTER BUFFER POINTER

MOV (R0)+,R2 ;R2 HAS ADDRESS OF ERROR REG
MOV #BIT0,R1 ;R1 WILL HAVE BIT UNDER TEST
BIS #CLR,@RHCS2 ;CLEAR RHCS2
MOV UNIT,@RHCS2 ;REINSTATE UNIT NO.
MOV R1,@R2 ;SET ERROR BIT
JSR PC,PUTREG ;READ AND SAVE REGISTERS
CMPB R5,AS ;ONLY THE BIT IN R5 SHOULD BE
;SET IN RHAS
BEQ 3$ ;LOOK @ RHDS1 IF GOOD
EMT 20

;ERROR BIT IN AN ERROR
;REGISTER, THE CORRESPONDING
;RHAS BIT DID NOT SET

MOV DS1,-(SP) ;GET RHDS1
BIC #VV!PROG,(SP) ;REMOVE VV AND PROG
    
```

```

54 030466 022726 140600      CMP      #ATA!ERR!DPR!DRY,(SP)+;THESE BITS PLUS VV SHOULD BE IN RHDS1
55 030472 001401      BEQ      4$      ;CHECK 'GO' NEXT, IF THIS WAS OK
56 030474 104020      EMT      20
57
58
59
60
61
62 030476 012777 000001 150534 4$:      MOV      #GO,@RHCS1      ;GIVE NO-OP
63 030504 004737 041064      JSR      PC,PUTREG      ;SAVE REGISTERS
64 030510 020112      CMP      R1,@R2      ;GO SHOULD NOT CLEAR ERROR
65 030512 001410      BEQ      5$      ;FURTHER CHECK OF 'GO' FUNCTIONALITY
66 030514 010237 041130      MOV      R2,REGADR      ;FAILING REGISTER
67 030520 010137 001124      MOV      R1,$GDDAT      ;GOOD DATA
68 030524 013737 001312 001126      MOV      CS2,$BDDAT      ;BAD DATA
69 030532 104001      EMT      1
70
71
72 030534 013746 001336      5$:      MOV      DS1,-(SP)      ;GET RHDS1
73 030540 042716 001100      BIC      #VV!PROG,(SP)      ;CLEAR VV AND PROG
74 030544 022726 140600      CMP      #ATA!ERR!DPR!DRY,(SP)+;GO SHOULD NOT CLEAR ANY BITS
75 030550 001401      BEQ      7$      ;CHECK NEXT ERROR BIT IF A-OK
76 030552 104020      EMT      20
77
78
79
80
81 030554 006301      7$:      ASL      R1      ;GET NEXT BIT TO THE LEFT
82 030556 103403      BCS      10$      ;GO ON TO NEXT REGISTER IF DONE
83 030560 031001      BIT      (R0),R1      ;IS THIS BIT TO BE TESTED ?
84 030562 001374      BNE      7$      ;IF NOT, GET NEXT ONE
85 030564 000717      BR       2$      ;IF TO BE TESTED, GO DO IT !
86
87 030566 005720      10$:     TST      (R0)+      ;ADVANCE R0 TO NEXT ERROR REG.
88 030570 005337 001200      DEC      $TMP1      ;REGISTER COUNTER
89 030574 001310      BNE      1$      ;DO NEXT ONE, IF 3 NOT COMPLETE
90
91
92
93
94
95 030576 004737 041460      11$:     JSR      PC,CLDISK      ;CLEAR
96 030602 012737 030576 001110      MOV      #11,$LPERR      ;ERROR RETURN
97 030610 012714 177777      MOV      #-1,@R4      ;SET BIT IN RHAS AND ATA IN RHDS1
98 030614 013777 001416 150434      MOV      ATTENT,@RHAS      ;WRITE 1 INTO DRIVE BIT POSITION
99 030622 004737 041064      JSR      PC,PUTREG      ;SAVE REGISTERS
100 030626 105737 001332      TSTB     AS      ;THIS SHOULD BE ZERO
101 030632 001401      BEQ      12$
102 030634 104020      EMT      20
103
104
105
106 030636 013746 001336      12$:     MOV      DS1,-(SP)      ;GET RHDS1
107 030642 042716 001000      BIC      #PROG,(SP)      ;MASK PROGRAMABLE
108 030646 022726 040700      CMP      #ERR!VV!DPR!DRY,(SP)+
109
110

```

```

;AT THE DRIVE BIT POSITION
;DID NOT CLEAR IT

```

```

;RHDS1 SHOULD HAVE THESE BITS
;BUT ATA SHOULD BE CLEARED

```

```

111 030652 001401      BEQ    13$      ;CHECK RHER1 IF GOOD
112 030654 104020      EMT    20
113                                     ;DRIVE BIT POSITION DID NOT
114                                     ;CLEAR ATA IN RHDS1
115
116 030656 022737 177777 001316 13$:  CMP    #-1,ER1 ;RHER1 SHOULD NOT CHANGE
117                                     ;BY CLEARING RHAS
118 030664 001401      BEQ    TST72    ;BRANCH IF GOOD
119 030666 104020      EMT    20
120                                     ;RHAS BY MOVING '1' INTO
121                                     ;THE DRIVE BIT POSITION
122
130
  
```

```

:*****
:*TEST 72      BUS ADDRESS INHIBIT
:*READ CYLINDER0, FORMAT 16 BITS PER WORD
:*TRACK0, SECTOR 1, KEYS 0, 10 WORDS OF 177400
:*THIS IS DONE WITH BUS ADDRESS INHIBIT SET
:*ANY ERROR LOGIC INDICATION IS NOT CONCLUSIVE ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT BEEN CHECKED YET (ON FIRST PASS)
:*****
TST72: SCOPE
  
```

```

131 030670 000004
132 030672 012706 001100      MOV    #STACK,SP      ;RESET STACK
133 030676 012737 000072 001432  MOV    #72,TSTNM      ;MOVE #72 TO TEST NUMBER
134 030704 004037 041376      JSR    R0,CLAREA      ;CLEAR SIMULATED DISK
135 030710 050224              .WORD  DISK            ;FROM
136 030712 051222              .WORD  DISK+776        ;TO
137 030714 177400              .WORD  177400          ;DATA
138
139 030716 004037 041376      JSR    R0,CLAREA      ;CLEAR READ INTO BUFFER
140 030722 002554              .WORD  REINTO          ;FROM
141 030724 003552              .WORD  REINTO+776      ;TO
142 030726 000000              .WORD  0               ;DATA
143
144                                     ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
145
146 030730 012737 010000 046306  MOV    #FMT22,CYL     ;CYLINDER 0 16 BITS PER WORD FORMAT
147 030736 105037 046311      CLRB   SECOTR+1       ;TRACK 0
148 030742 112737 000001 046310  MOVB   #1,SECOTR      ;SECTOR 1
149 030750 005037 046312      CLR    KEY1           ;KEY1=0
150 030754 005037 046314      CLR    KEY2           ;KEY2=0
151 030760 012737 000012 046366  MOV    #10,,DAWORD    ;NO. OF DATA WORDS
152 030766 005037 046316      CLR    X              ;THIS IS A READ COMMAND
153 030772 004537 042666      JSR    R5,CRC         ;GO TO CALCULATE CRC
154 030776 046306      CYL
155 031000 050206      WCRC
156
157                                     ;THESE ARE REGULAR SETUPS
158
159 031002 004737 041460      JSR    PC,CLDISK      ;SETUP GENERAL REGISTERS
160 031006 013711 001466      MOV    READAT,@R1     ;READ DATA INTO RHCS1=70
161 031012 012777 177766 150212  MOV    #-10,@RHWC      ;10 DATA WORDS
162 031020 012777 002554 150206  MOV    #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
163 031026 112746 000001      MOVB   #1,-(SP)       ;IN LOWER BYTE GET SECTOR 1
164 031032 112766 000000 000001  MOVB   #0,1(SP)        ;GET TRACK0 IN UPPER BYTE
165 031040 012677 150200      MOV    (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
166 031044 012777 014000 150176  MOV    #FMT22!ECI,@RHOF ;16 BITS PER WORD
  
```

```

167                                     ;ECC CORRECTION INHIBIT BECAUSE
168                                     ;ECC IS NOT CHECKED HERE
169 031052 005077 150174                CLR    @RHCA
170 031056 004737 041514                JSR    PC,CHECKT
    031062 104401 062726                TYPE   ,CPHALT
                                     ;CYLINDER 0
                                     ;CHECK THAT DVA,RDY,DPR,DRY = 1
                                     ;AND THAT NO OTHERS = 1. CANNOT CON-
                                     ;TINUE TESTING IF BOTH AREN'T TRUE
    031066 000000                HALT
171 031070 052777 000010 150140        BIS    #BAI,@RHCS2
172 031076 005037 001406                CLR    ERFLG$ ;CLEAR ERROR FLAG
173 031102 004737 046146                JSR    PC,COMHD ;READ DATA
174
175                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUT
176                                     ;FROM "COMHD" ROUTINE IN MEANS DATA IS TO BE CHECKED
177
178                                     ;NOW THE DATA READ INTO "REINTO" BUFFER WILL
179                                     ;BE CHECKED, ONLY ONE WORD SHOULD BE CHANGED
180                                     ;ALL OTHER WORDS SHOULD REMAIN UNCHANGED
181                                     ;THE "WRFROM" BUFFER IS FILLED WITH EXPECTED DATA AND CHECKED
182
183 031106 005037 001406                CLR    ERFLG$ ;CLEAR FLAG
184 031112 004037 041376                JSR    R0,CLAREA ;CLEAR BUFFER
185 031116 001510                WRFROM ;FROM
186 031120 002506                WRFROM+776 ;TO
187 031122 000000                0 ;DATA
188
189                                     ;EXPECTED DATA IS 177400 IN FIRST LOCATION ONLY
190 031124 012737 177400 001510        MOV    #177400,WRFROM ;EXPECTED DATA
191
192                                     ;NOW READ DATA BUFFER IS CHECKED
193
194 031132 012700 001510                MOV    #WRFROM,R0 ;GOOD DATA
195 031136 012701 002554                MOV    #REINTO,R1 ;DATA READ
196 031142 012702 000400                MOV    #256.,R2 ;COUNTER
197 031146 012737 000401 046426 1$:    MOV    #257.,ERWORD ;FOR ERROR WORD NO
198 031154 022021                CMP    (R0)+,(R1)+ ;COMPARE GOOD WITH READ BUFFER
199 031156 001424                BEQ    2$ ;BRANCH IF GOOD
200 031160 014037 001124                MOV    -(R0),SGDDAT ;GOOD DATA
201 031164 014137 001126                MOV    -(R1),SBDDAT ;BAD DATA
202 031170 160237 046426                SUB    R2,ERWORD ;ERROR WORD NO
203 031174 005737 001406                TST    ERFLG$ ;ANY ERRORS ALREADY THERE
204 031200 001002                BNE    3$ ;IF YES BRANCH DO NOT TYPE HEADER
205 031202 104004                EMT    4
206 031204 000401                BR     4$ ;BRANCH TO AVOID PRINTING NEXT ERROR
207 031206 031206 104005                3$:   EMT    5
208
209                                     ;WORDS
210                                     ;WORD NOS 11-256 HAVE NOT BEEN
211                                     ;READ AND BUFFER SHOULD BE
212                                     ;ZERO IF OTHER THAN ZERO
213                                     ;WRONG NUMBER OF WORDS HAVE
214                                     ;BEEN READ IN THE DISK NOW
215                                     ;CONTAINS 177400 ALL 256
216                                     ;WORDS BUT ONLY 10 WORDS
217                                     ;SHOULD BE READ IN
218 031210 022021 147722                4$:   CMP    (R0)+,(R1)+ ;UNDO -(R0) AND -(R1) FOR ERROR
219 031212 017746                MOV    @SWR,-(SP) ;GET SWITCH SETTING
    
```

```

220 031216 042716 177177      BIC    #177177,(SP)    ;KEEP ONLY SWITCH 7 AND 8
221 031222 022726 000200      CMP    #SW07,(SP)+    ;IS 7 SET AND 8 RESET
222 031226 001402              BEQ    TST73           ;BRANCH OUT IF YES
223 031230 005302              2$:   DEC    R2         ;COUNT
224 031232 001345              BNE    1$             ;BRANCH IF NOT COMPLETE
225
233
;*****
;*TEST 73      RHCS2 - BIT # 11 - NEM
;*READ CYLINDERO, FORMAT 16 BITS PER WORD
;*TRACK0, SECTOR 1, KEYS 0, 1 WORD OF 177400
;*THIS IS DONE WITH BUS ADDRESS INHIBIT SET
;*BUS ADDRESS USED IS 760000 THIS IS ALWAYS NON EXISTANT
;*THIS SHOULD SET NEM
;*****
234 031234 000004              TST73: SCOPE
235 031236 012706 001100      MOV    #STACK,SP     ;RESET STACK
236 031242 012737 000073 001432  MOV    #73,TSTNM     ;MOVE #73 TO TEST NUMBER

;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

031250 005737 001436          TST    RH70          ;TEST FLAG FOR RH70 CONTROLLER
031254 001402                BEQ    64$           ;IF FLAG = 1, THIS TEST IS SKIPPED
031256 000137 031600          JMP    TST74         ;JUMP TO NEXT TEST
031262
64$:
;IF FLAG = 0, DO THIS TEST

237
238 031262 004037 041376      JSR    R0,CLAREA     ;CLEAR SIMULATED DISK
239 031266 050224              .WORD  DISK          ;FROM
240 031270 051222              .WORD  DISK+776     ;TO
241 031272 177400              .WORD  177400       ;DATA
242
243
244
245
;THESE ARE TO SETUP FOR DISKLESS USE ONLY
246 031274 012737 010000 046306  MOV    #FMT22,CYL    ;CYLINDER 0, 16 BITS PER WORD FORMAT
247 031302 105037 046311      CLR    SECOTR+1     ;TRACK 0
248 031306 112737 000001 046310  MOVB  #1,SECOTR     ;SECTOR 1
249 031314 005037 046312      CLR    KEY1         ;KEY1=0
250 031320 005037 046314      CLR    KEY2         ;KEY2=0
251 031324 012737 000001 046366  MOV    #1.,DAWORD   ;NO. OF DATA WORDS
252 031332 005037 046316      CLR    X            ;THIS IS A READ COMMAND
253 031336 004537 042666      JSR    R5,CRC       ;GO TO CALCULATE CRC
254 031342 046306              CYL
255 031344 050206              WCRC
256
257
258
;THESE ARE REGULAR SETUPS
259 031346 004737 041460      JSR    PC,CLDISK    ;SETUP GENERAL REGISTERS
260 031352 013711 001466      MOV    READAT,@R1   ;READ DATA INTO RHCS1=70
261 031356 012777 177777 147646  MOV    #-1.,@RHWC   ;10 DATA WORDS
262 031364 012777 160000 147642  MOV    #160000,@RHBA ;STARTING ADDRESS OF READ BUFFER
263 031372 052711 001400      BIS    #A16!A17,@R1 ;IS 760000
264 031376 112746 000001      MOVB  #1,-(SP)     ;IN LOWER BYTE GET SECTOR 1
265 031402 112766 000000 000001  MOVB  #0,1(SP)     ;GET TRACK0 IN UPPER BYTE
266 031410 012677 147630      MOV    (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
267 031414 012777 014000 147626  MOV    #FMT22!ECI,@RHOF ;16 BITS PER WORD
268
269
;ECC CORRECTION INHIBIT BECAUSE
;ECC IS NOT CHECKED HERE
    
```

```

270 031422 005077 147624 CLR @RHCA ;CYLINDER 0
271 031426 004737 041514 JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
031432 104401 062726 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
031436 000000 HALT ;STOP THE TEST
272 031440 052777 000010 147570 BIS #BAI,@RHCS2 ;SET BUS ADDRESS INHIBIT
273 031446 005037 001406 CLR ERFLGS ;CLEAR ERROR FLAG
274 031452 004737 046146 JSR PC,COMHD ;READ DATA
275
276
277
278 031456 011137 001126 1$: MOV @R1,$BDDAT ;TEST DATA
279
280 031462 022737 145670 001126 CMP #SC!TRE!DVA!A16!A17!RDY!70,$BDDAT ;COMPARE RHCS1
281 031470 001406 BEQ 2$ ;BRANCH IF GOOD
282 031472 012737 144270 001124 MOV #SC!TRE!DVA!RDY!70,$GDDAT ;GOOD DATA
283 031500 010137 041130 MOV R1,REGADR ;REGISTER RHCS1
284 031504 104001 EMT 1
285 ;MEMORY DID NOT SET
286 ;REQUIRED BITS
287 031506 013746 001374 2$: MOV UNIT,-(SP) ;GET UNIT NUMBER
288 031512 052716 004110 BIS #NEM!IR!BAI,(SP) ;INCLUDE NEM BAI AND IR
289 031516 012637 001124 MOV (SP)+,$GDDAT
290 031522 011237 001126 MOV @R2,$BDDAT ;TEST DATA
291 031526 023737 001124 001126 CMP $GDDAT,$BDDAT;COMPARE RHCS2
292 031534 001403 BEQ 3$
293 031536 010237 041130 MOV R2,REGADR ;REGISTER ADDRESS
294 031542 104001 EMT 1
295 ;CAUSED AN ERROR SHOULD SET NEM
296 031544 017737 147464 001126 3$: MOV @RHBA,$BDDAT ;TEST DATA
297
298 031552 022737 160000 001126 CMP #160000,$BDDAT ;COMPARE RHBA
299 031560 001407 BEQ 4$ ;BRANCH IF GOOD
300 031562 012737 160000 001124 MOV #160000,$GDDAT;GOOD DATA
301 031570 013737 001234 041130 MOV RHBA,REGADR ;REGISTER ADDRESS RHBA
302 031576 104001 EMT 1
303 ;RHBA DOES NOT HAVE 160002
304 031600 4$:
305
315

```

```

:*****
:*TEST 74 WRITE CHECK ERROR
:*WRITE CHECK DATA CYLINDER 0 FORMAT 16 BITS PER WORD
:*TRACK 1, SECTOR 1, KEYS 0, 32 WORDS OF DATA
:*FIFTH WORD IS CHANGED ON DISK TO GIVE WRITE CHECK ERROR
:*ANY DEVICE LOGIC ERROR INDICATIONS ARE NOT CONCLUSIVE
:*ON FIRST PASS
:*BECAUSE ERROR LOGIC HAS NOT YET BEEN CHECKED
:*ONLY RH WRITE CHECK ERROR IS TESTED
:*****
TST74: SCOPE

```

```

031600 000004
316
317
318 ;DATA TABLE
319 ;TOTAL OF 32 WORDS CONSISTING OF
320 ;16 WORDS OF FLOATING ONES (EG. 1, 2, 4, 10)
321 ;16 WORDS OF FLOATING ZEROS (EG. 177776, 177775)
322 031602 012706 001100 MOV #STACK,SP ;RESET STACK

```

```

323 031606 012737 000074 001432      MOV    #74,TSTNM      ;MOVE #74 TO TEST NUMBER
324 031614 004737 041460              JSR    PC,CLDISK     ;INIT AND SET UP GENERAL REGISTERS
325
326                                ;SET UP 'REINTO' FOR WHAT IS TO BE READ
327
328 031620 012700 0000C1              MOV    #1,R0         ;GETTING READY TO FLOAT 1
329 031624 012701 002554              MOV    #REINTO,R1    ;STARTING ADDRESS WHERE 1 GOES
330 031630 010021 000000      1$:   MOV    R0,(R1)+      ;MOVE FLOATING 1
331 031632 006100              ROL    R0             ;GET 1 ONE BIT LEFT
332 031634 103375              BCC    1$            ;BRANCH IF 16 NOT DONE
333 031636 012700 177776              MOV    #177776,R0    ;GETTING READY TO FLOAT 0
334 031642 012701 002614              MOV    #REINTO+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
335 031646 010021 000000      2$:   MOV    R0,(R1)+      ;MOVE IN FLOATING 0
336 031650 000261              SEC                     ;SET CARRY
337 031652 006100              ROL    R0             ;GET 0 ONE BIT LEFT
338 031654 103774              BCS    2$            ;BRANCH IF 16 NOT DONE
339
340 031656 004037 041376              JSR    R0,CLAREA     ;FILL REST OF BUFFER WITH 1
341 031662 002654              .WORD REINTO+<32.*2> ;FROM
342 031664 003552              .WORD REINTO+776     ;TO
343 031666 000001              .WORD 1               ;WITH DATA
344
345                                ;SET UP SIMULATED DISK WITH WHAT IS TO BE READ
346
347 031670 012700 000001              MOV    #1,R0         ;GETTING READY TO FLOAT 1
348 031674 012701 050224              MOV    #DISK,R1      ;STARTING ADDRESS WHERE 1 GOES
349 031700 010021 000000      3$:   MOV    R0,(R1)+      ;MOVE FLOATING 1
350 031702 006100              ROL    R0             ;GET 1 ONE BIT LEFT
351 031704 103375              BCC    3$            ;BRANCH IF 16 NOT DONE
352
353 031706 012700 177776              MOV    #177776,R0    ;GETTING READY TO FLOAT 0
354 031712 012701 050264              MOV    #DISK+<16.*2>,R1 ;STARTING ADDRESS WHERE 177776 GOES
355 031716 010021 000000      4$:   MOV    R0,(R1)+      ;MOVE FLOATING 0
356 031720 000261              SEC                     ;SET CARRY
357 031722 006100              ROL    R0             ;GET 0 ONE BIT LEFT
358 031724 103774              BCS    4$            ;BRANCH IF 16 NOT DONE
359
360 031726 004037 041376              JSR    R0,CLAREA     ;FILL REST OF BUFFER WITH 0
361 031732 050324              .WORD DISK+<32.*2>  ;FROM
362 031734 051222              .WORD DISK+776     ;TO
363 031736 000000              .WORD 0               ;WITH DATA
364
365                                ;CHANGE FIFTH WORD TO 0 ON DISK
366
367 031740 005037 050234              CLR    DISK+10      ;CLEAR FIFTH WORD ON DISK
368 031744 005037 001406              CLR    ERFLGS      ;CLEAR ERROR FLAG
369 031750 004737 042516              JSR    PC,WRCHDA     ;WRITE CHECK DATA
370                                ;CYLINDER 0, TRACK 1, SECTOR 1
371                                ;KEYS 0, 32 WORDS.
372
373                                ;IF THE PROGRAM COMES BACK HERE THEN WRITE CHECK
374                                ;HAS BEEN COMPLETED, NOW WRITE CHECK ERROR BIT IS TESTED
375                                ;ALONG WITH RHWC FOR PROPER WORD COUNT AND RHBA FOR ADDRESS
376
377 031754 013746 001374              MOV    UNIT,-(SP)    ;GET UNIT NUMBER
378 031760 052716 040300              BIS    #IR!OR!WCE,(SP) ;ONLY BIT 6 SHOULD BE SET
379 031764 004737 041064              JSR    PC,PUTREG     ;SAVE REGISTERS

```

```

380 031770 022637 001312      CMP      (SP)+,CS2      :COMPARE RHCS2
381 031774 001407              BEQ      6$             :BRANCH IF GOOD
382 031776 032737 040000 001312  BIT      #WCE,CS2      :WRITE CHECK ERROR HIGH?
383 032004 001002              BNE      5$             :BRANCH IF ERROR NOT DUE TO 'WCE'
384 032006 104017              EMT      17
385 032010 000401              BR       6$             :RHBA CONTAINS ADDRESS+2
386                                :OF THE WORD IN MEMORY FROM
387                                :THE DISK THAT DID NOT COMPARE
388                                :TRE AND SC WILL BE SET DUE TO WCE
389 032012              5$:              EMT      17
    032012 104017
390                                :BUT SOME BITS OTHER THAN
391                                :IR AND UNIT NO. WERE SET
392
393 032014 005737 001436      6$:      TST      RH70      :TEST FOR RH70 CONTROLLER
394 032020 001414              BEQ      16$           :SKIP RH70 CODE AND DO RH11 IF NOT
395
396 032022 022737 177750 001306  CMP      #-24.,WC      :COMPARE RHWC AFTER A FORCED
397                                :WRITE CHECK ERROR
398 032030 001402              BEQ      17$           :CHECK RHBA IF GOOD
399 032032 104017              EMT      17
400                                :FORCED WRITE CHECK ERROR ON FIFTH WORD
401 032034 000421              BR       15$           :BRANCH TO CONTINUE TEST
402
403 032036 022737 002574 001310 17$:      CMP      #REINTO+<8.*2>,.BA :COMPARE RHBA AFTER A FORCED
404                                :WRITE CHECK ERROR IN FIFTH WORD
405 032044 001415              BEQ      15$           :CONTINUE IF GOOD
406 032046 104017              EMT      17
407                                :FORCED WRITE CHECK ERROR ON FIFTH WORD
408 032050 000413              BR       15$           :SKIP RH11 CODE AND CONTINUE WITH TEST
409
410 032052 022737 177745 001306 16$:      CMP      #-27.,WC      :COMPARE RHWC AFTER A FORCED
411                                :WRITE CHECK ERROR
412 032060 001402              BEQ      14$           :CHECK RHBA IF GOOD
413 032062 104017              EMT      17
414                                :FORCED WRITE CHECK ERROR ON FIFTH WORD
415 032064 000405              BR       15$           :BRANCH TO CONTINUE TEST
416
417 032066 022737 002566 001310 14$:      CMP      #REINTO+<5.*2>,.BA :COMPARE RHBA AFTER FORCED
418                                :WRITE CHECK ERROR IN FIFTH WORD
419 032074 001401              BEQ      15$           :CONTINUE IF GOOD
420 032076 104017              EMT      17
421                                :FORCED WRITE CHECK ERROR ON FIFTH WORD
422
423                                :NOW CHECK MEMORY TO SEE IF ANYTHING GOT DESTROYED
424                                :FILL 'WRFROM' WITH WHAT SHOULD BE IN REINTO THEN CHECK
425
426 032100 005037 001406      15$:      CLR      ERFLG$      ;CLEAR ERROR FLAG
427 032104 012700 000001      MOV      #1,R0         ;GETTING READY TO FLOAT 1
428 032110 012701 001510      MOV      #WRFROM,R1    ;START ADDRESS WHERE 1 GOES
429 032114 010021              7$:      MOV      R0,(R1)+      ;MOVE FLOATING 1
430 032116 006100              ROL      R0             ;GET 1 ONE BIT LEFT
431 032120 103375              BCC      7$            ;BRANCH IF 16 NOT DONE
432
433 032122 012700 177776      MOV      #177776,R0    ;GETTING READY TO FLOAT 0
434 032126 012701 001550      MOV      #WRFROM+<16.*2>,.R1 ;STARTING ADDRESS WHERE 177776 GOES
435 032132 010021              10$:     MOV      R0,(R1)+      ;MOVE IN FLOATING 0
    
```



```

436 032134 000261          SEC          :SET CARRY
437 032136 006100          ROL      RO          :GET 0 ONE BIT LEFT
438 032140 103774          BCS      10$         :BRANCH IF CARRY SET
439
440 032142 004037 041376   JSR      RO,CLAREA    :FILL REST OF BUFFER WITH 1
441 032146 001610          .WORD   WRFROM+<32.*2> :FROM
442 032150 002506          .WORD   WRFROM+776     :TO
443 032152 000001          .WORD   1              :WITH DATA
444
445                          :NOW THE READ BUFFER WILL BE CHECKED
446
447 032154 004037 042354   JSR      RO,COMPAR    :CHECK
448 032160 001510          WRFROM                    :GOOD BUFFER
449 032162 002554          REINTO                    :TEST BUFFER
450 032164 000400          256.                      :NUMBER OF WORDS CHECKED
451 032166 032174          11$                      :RETURN POINT FOR ERROR HEADER
452 032170 032200          12$                      :RETURN POINT FOR ERROR DATA
453
454 032172 032206          TST75                      :RETURN FOR GOOD COMPARISON
455
456 032174          11$:
457 032176 104004          EMT      4
458 032176 000207          RTS      PC              :RETURN TO COMPARISON SUBROUTINE
459 032200          12$:
460 032200 104005          EMT      5
461
462
463 032202 000207          RTS      PC              :CHANGED AFTER A WRITE
464
465 032204 000240          :CHECK DATA COMMAND
466
467
468
469
470
471
472
473
474
475 032206 000004          :WORD NO CONTAINS THE WORD
476 032210 012706 001100   :NUMBER THAT GOT CHANGED
477 032214 012737 000075 001432 :RETURN TO COMPARISON SUBROUTINE
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
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
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
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
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

```

*****
*TEST 75 ERROR REGISTER #1-BIT 4 -FORMAT ERROR
*THE SIMULATED DISK IS FILLED WITH CYLINGER 0 TRACK 1
*SECTOR 0 FORMAT=18 BITS PER WORD AND 4 WORDS
*OF 125252, A READ HEADER AND DATA COMMAND IS GIVEN WITH 16 BITS
*PER WORD FORMAT, FER=BIT4 SHOULD SET BUT THE
*READ SHOULD BE COMPLETE
*****

```

```

TST75: SCOPE
MOV #STACK,SP :RESET STACK
MOV #75,TSTNM :MOVE #75 TO TEST NUMBER

:SETUP FOR WHAT IS TO BE READ
:HEADER CRC IS RESTORED FROM A SUBROUTINE

MOV #125252, -(SP) :DATA TO BE READ
MOV #256.,R5 :COUNTER
MOV #DISK,RO :START OF SIMULATED DISK DATA
1$: MOV (SP),(RO)+ :MOVE IN DATA ON TO SIMULATED DISK
DEC R5 :COUNT
BNE 1$ :BRANCH IF 256 NOT COMPLETE
TST (SP)+ :UNDO -(SP)
MOV #17.,R5 :2 ECC WORDS
:1 DATA GAP

```

```

032252 005020      2$: CLR      (R0)+      ;14 TOLERANCE GAP
032254 005305      DEC      R5          ;CLEAR ECC, DATA GAP, AND
032256 001375      BNE      2$         ;TOLERANCE GAP
                                ;BRANCH IF NOT COMPLETE
  
```

;THESE ARE TO SETUP FOR DISKLESS USE ONLY

```

032260 012737 000000 046306      MOV      #0!0,CYL      ;16 BITS PER WORD
                                ;CYLINDER 0, FORMAT 16 BITS
032266 112737 000001 046311      MOVB     #1,SECOTR+1   ;TRACK 1
032274 112737 000000 046310      MOVB     #0,SECOTR     ;SECTOR 0
032302 012737 000000 046312      MOV      #0,KEY1 ;KEY1=0
032310 012737 000000 046314      MOV      #0,KEY2 ;KEY2=0
032316 012737 000004 046366      MOV      #4.,DAWORD   ;NO. OF DATA WORDS
032327 005037 046316              CLR      X            ;THIS IS A READ COMMAND
032330 004537 042666              JSR      R5,CRC       ;GO TO CALCULATE CRC
032334 046306              CYL
032336 050206              WCRC
  
```

;THESE ARE REGULAR SETUPS FOR RH11 AND 'REINTO'

```

032340 004737 041460              JSR      PC,CLDISK    ;SETUP GENERAL REGISTERS
032344 012777 177770 146660      MOV      #-4.-4.,@RHWC ;4. DATA 4 HEADER WORDS
032352 012777 002554 146654      MOV      #REINTO,@RHBA ;STARTING ADDRESS OF READ BUFFER
032360 112746 000000              MOVB     #0,-(SP)     ;IN LOWER BYTE GET SECTOR
032364 112766 000001 000001      MOVB     #1,1(SP)    ;GET TRACK IN HIGHER BYTE
032372 012677 146646              MOV      (SP)+,@RHDST ;TRACK/SECTOR IN RHDST
032376 012777 014000 146644      MOV      #FMT22!ECI,@RHOF ;16 BITS PER WORD
                                ;ECC CORRECTION INHIBIT
                                ;BECAUSE ECC IS NOT GOING
                                ;TO BE CHECKED
                                ;CYLINDER 0
032404 005077 146642              CLR      @RHCA
032410 004737 041514              JSR      PC,CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
032414 104401 062726              TYPE     ,CPHALT    ;AND THAT NO OTHERS = 1. CANNOT CON-
                                ;TINUE TESTING IF BOTH AREN'T TRUE
032420 000000              HALT              ;STOP THE TEST
032422 013711 001470              MOV      REFOR,@R1   ;READ HEADER AND DATA=72
032426 005037 001406              CLR      ERFLG$     ;CLEAR ERROR FLAG
032432 004737 046146              JSR      PC,COMHD    ;READ HEADER AND DATA
  
```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 ;FROM THE 'COMHD' ROUTINE THAT MEANS SECTOR GAP,
 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
 ;DETECTED.

;RHWC IS CHECKED TO BE = 0 AFTER THE READ OPERATION

```

032436 017737 146570 001126      MOV      @RHWC,$BDDAT ;LOAD AND TEST RHWC
032444 001401              BEQ      20$         ;SHOULD = 0
032446 104040              EMT      40
                                ;HEADER AND DATA
  
```

;HEADER AND DATA ARE TO BE CHECKED.

:IN CHECKING READ DATA THE WRITE FROM BUFFER
: 'WRFROM' IS FILLED WITH EXPECTED DATA AND
: COMPARISONS ARE MADE.

032450	005737	001406	20\$:	TST	ERFLG\$:ANY ERRORS ALREADY THERE
032454	001055			BNE	TST76	:BRANCH IF YES
032456	004737	041704		JSR	PC,CHECKE	:CHECK THAT BITS = 1
032462	104401	062726		TYPE	,CPHALT	:CANNOT CONTINUE TESTING IF THEY DON'T
032466	000000			HALT		:STOP THE TEST
032470	012700	001510		MOV	#WRFROM,R0	:GETTING READY TO FILL EXPECTED DATA
032474	012720	000000		MOV	#0,(R0)+	:CYLINDER 0
032500	112746	000000		MOVB	#0,-(SP)	:IN LOWER BYTE GET SECTOR
032504	112766	000001	000001	MOVB	#1,1(SP)	:GET TRACK IN HIGHER BYTE
032512	012620			MOV	(SP)+,(R0)+	:GET TRACK/SECTOR IN BUFFER
032514	012720	000000		MOV	#0,(R0)+	:KEY1 IN BUFFER
032520	012720	000000		MOV	#0,(R0)+	:KEY2 IN BUFFER
032524	012701	000400		MOV	#256,R1	:DATA WORD COUNTER
032530	012702	125252		MOV	#125252,R2	:DATA
032534	010220		3\$:	MOV	R2,(R0)+	:DATA INTO BUFFER
032536	005301			DEC	R1	:COUNT
032540	001375			BNE	3\$:BRANCH IF 256 NOT DONE

:NOW READ DATA BUFFER WILL BE CHECKED

032542	004037	042354		JSR	R0,COMPAR	:CHECK
032546	001510			WRFROM		:GOOD BUFFER
032550	002554			REINTO		:TEST BUFFER
032552	000010			4+4.		:NUMBER OF WORDS CHECKED
032554	032562			4\$:RETURN POINT FOR ERROR HEADER
032556	032566			5\$:RETURN POINT FOR ERROR DATA
032560	032572			6\$:RETURN FOR GOOD COMPARISON
032562			4\$:			
032562	104004			EMT	4	
032564	000207			RTS	PC	:RETURN TO 'COMPAR'
032566			5\$:			
032566	104005			EMT	5	
						:HEADER WORDS
						:5 TO 260 ARE DATA WORDS
032570	000207			RTS	PC	:RETURN TO 'COMPAR'

:NOW SEE THAT FORMAT ERROR BIT GOT SET

478							
479							
480							
481	032572	004737	041064	6\$:	JSR	PC,PUTREG	:SAVE REGISTERS
482							
483	032576	022737	100020	001316	CMP	#FER!DCK,ER1	:FORMAT ERROR SHOULD BE SET
484	032604	001401			BEQ	TST76	:BRANCH IF GOOD
485	032606	104020			EMT	20	
486							:WHEN THE DISK HAD
487							:THE FORMAT BIT=0= 18 BITS PER
488							:WORD THE READ WAS
489							:COMPLETED BUT ERROR REG
490							:WAS NOT RIGHT
491							:NOTE DCK WILL BE SET BECAUSE
492							:ECC HAS NOT BEEN GENERATED
493							
500							

:*****

:*TEST 76 ERROR REGISTER #1-BIT 4 -FORMAT ERROR
:*THE SIMULATED DISK HEADER IS FILLED WITH CYLINDER 0
:*TRACK 0, SECTOR 0 FORMAT 18 BITS PER WORD
:*A WRITE DATA COMMAND IS GIVEN WITH SAME HEADER
:*EXCEPT FORMAT BIT. THE DATA SHOULD NOT BE WRITTEN.
:*****

032610 000004

TST76: SCOPE

501
502
503
504
505 032612 012706 001100
506 032616 012737 000076 001432
507
508 032624 012737 177777 046422
509
510 032632 004037 041376
511 032636 050224
512 032640 051250
513 032642 000000
514
515
516

:NOW A WRITE DATA WILL BE ATTEMPTED WITH
:WRONG FORMAT BIT

MOV #STACK,SP ;RESET STACK
MOV #76,TSTNM ;MOVE #76 TO TEST NUMBER
MOV #-1,NOSYNC ;SET FLAG SO THAT DATA SYNC
;AND DATA IS NOT READ
FRMAT1: JSR R0,CLAREA ;CLEAR SIMULATED DISK
.WORD DISK ;FROM
.WORD TOLGAP+16 ;TO
.WORD 0 ;DATA

:THESE ARE SETUP FOR DISKLESS USE ONLY

517 032644 005037 046306
518 032650 105037 046311
519 032654 105037 046310
520 032660 005037 046312
521 032664 005037 046314
522 032670 012737 000004 046354
523 032676 012737 000001 046316
524 032704 004537 042666
525 032710 051444
526 032712 051454
527
528
529

CLR CYL ;CYLINDER 0, FORMAT 18 BIT WORDS
CLRB SECOTR+1 ;TRACK 0
CLRB SECOTR ;SECTOR 0
CLR KEY1 ;KEY1 0
CLR KEY2 ;KEY2 0
MOV #4,NOWORD ;NO OF DATA WORDS
MOV #1,X ;WRITE DATA
JSR R5,CRC ;GO TO CALCULATE CRC
WCYL
GCRC

:THESE AER REGULAR SETUPS

530 032714 004037 041376
531 032720 001510
532 032722 001516
533 032724 125252
534 032726 004737 041460
535 032732 012777 146272
536 032740 012777 001510 146266
537 032746 005077 146272
538 032752 012777 010000 146270
539 032760 005077 146266
540 032764 004737 041514
032770 104401 062726

JSR R0,CLAREA ;FILL WRITE FROM BUFFER WITH 125252
WRFROM ;FROM
WRFROM+6 ;TO
125252 ;DATA
JSR PC,CLDISK ;SETUP GENERAL REGISTERS
MOV #-4,@RHWC ;256 DATA WORDS
MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
CLR @RHDST ;TRACK=0 SECTOR=0
MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
CLR @RHCA ;CYLINDER 0
JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE

032774 000000
541 032776 013711 001462
542 033002 005037 001406
543 033006 004737 046146
544
545
546
547

HALT ;STOP THE TEST
MOV WRIDAT,@R1 ;WRITE DATA=60
CLR ERFLG\$;CLEAR ERROR FLAG
JSR PC,COMHD ;WRITE DATA

:IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
:FROM THE 'COMHD' ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
:HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY

```
548 ;AND SYNCs WERE CORRECTLY DETECTED
549 ;DATA IS TO BE CHECKED
550
551 033012 ( 74737 041064 JSR PC,PUTREG * ;SAVE REGISTERS
552 033016 005737 001406 TST ERFLG$ ;HAS ANY ERRORS OCCURED?
553 033022 001041 BNE 4$ ;BRANCH IF YES
554 033024 012700 000000 MOV #0,R0 ;GOOD DATA
555 033030 012701 050224 MOV #DISK,R1 ;DATA WRITTEN INTO 'DISK'
556 033034 012702 000004 MOV #4,R2 ;COUNTER
557 033040 012737 000005 046426 1$: MOV #5,ERWORD ;FOR ERROR WORD
558 033046 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
559 033050 001424 BEQ 3$ ;BRANCH IF GOOD
560 033052 010037 001124 MOV R0,$GDDAT ;GOOD DATA
561 033056 014137 001126 MOV -(R1),$BDDAT ;BAD DATA
562 033062 160237 046426 SUB R2,ERWORD ;ERROR WORD NO
563 033066 005737 001406 TST ERFLG$ ;ANY ERRORS ALREADY THERE?
564 033072 001002 BNE 2$ ;BRANCH IF YES
565 033074 104004 EMT 4
566 ;ON A WRITE DATA WITH
567 ;WRONG FORMAT NO DATA
568 ;SHOULD BE WRITTEN
569 ;WORD NO GIVES WORD IN ERROR
570 033076 000401 BR 5$ ;BRANCH TO AVOID PRINTING NEXT ERROR
571 033100 2$: EMT 5
572 033102 005721 5$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
573 033104 017746 146030 MOV @SWR,-(SP) ;GET SWITCH SETTING
574 033110 042716 177177 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
575 033114 022726 000200 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET.
576 033120 001402 BEQ 4$ ;BRANCH IF YES
577 033122 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
578 033124 001345 BNE 1$ ;BRANCH IF 256 NOT DONE
579
580 ;NOW CHECK TO SEE THAT FORMAT ERROR BIT GOT SET
581
582 033126 022737 000020 001316 4$: CMP #FER,ER1 ;FORMAT ERROR SHOULD BE SET
583 033134 001401 BEQ TST77 ;BRANCH IF GOOD
584 033136 104020 EMT 20
585 ;WAS ATTEMPTED WHEN THE DISK
586 ;HAD THE FORMAT BIT =0=18
587 ;BITS PER WORD THE WRITE
588 ;WAS CORRECTLY ABORTED
589 ;BUT ERROR REG. 1 WAS WRONG
590
604
```

```
::*****
:*TEST 77 RHER1 - BIT #2 - REG. MODIFICATION REFUSED
:*IN THIS TEST THE REGISTERS ARE IN TWO GROUPS
:*FIRST - RHCS1,RHDS1,RHOF,RHCA,RHER1,RHER2,RHER3 - SETS RMR
:*SECOND - RHMR,RHAS - DOES NOT SET RMR
:*IF WRITING IS ATTEMPTED DURING AN OPERATION
:*
:*ONLY ONE REGISTER IS WRITTEN INTO THAT IS RHCA
:*
:*1 THE REGISTERS CONTENTS ARE SAVED IN 'REINTO' BUFFER
:*2 WRITE HEADER AND DATA IS STARTED
:*3 ATTEMPT IS MADE TO WRITE INTO REGISTERS
:*4 ALL REGISTERS ARE COMPARED
```

```

*****
TST77: SCOPE
605 033140 000004          MOV      #STACK,SP      ;RESET STACK
606 033142 012706 001100  MOV      #77,TSTNM      ;MOVE #77 TO TEST NUMBER
607 033146 012737 000077 001432 JSR      PC,CLDISK      ;CLEAR DISK
608 033154 004737 041460 JSR      PC,CHECKT      ;CHECK THAT DVA,RDY,DPR,DRY = 1
033160 004737 041514 ;AND THAT NO OTHERS = 1. CANNOT CON-
033164 104401 062726 TYPE      ,CPHALT      ;TINUE TESTING IF BOTH AREN'T TRUE
;STOP THE TEST
609 033170 000000          HALT
610 033172 012700 001252  MOV      #RHCA,R0
611 033176 012005          MOV      (R0)+,R5      ;R5 HAS ADDRESS OF REG. UNDER TEST
612 033200 052777 000040 146030 1$:      MOV      #CLR,@RHCS2
613 033206 013777 001374 146022 2$:      BIS      #CLR,@RHCS2
        MOV      UNIT,@RHCS2      ;REINSTATE UNIT NO.
;SET UP FOR AN OPERATION (WRITE HEADER AND DATA)
616 033214 013777 001464 146016  MOV      WRIFOR,@RHCS1      ;WRITE HEADER AND DATA=62
617 ;IN RHCS1
618 033222 012777 177766 146002  MOV      #-10,@RHWC      ;10 WORDS
619 033230 012777 001510 145776  MOV      #WRFROM,@RHBA      ;BUS ADDRESS = WRFROM
620 033236 012777 000010 146000  MOV      #10,@RHDS1      ;DESIRED TRACK=0, SECTOR=10
621 033244 052777 000010 145764  BIS      #BAI,@RHCS2      ;BUS ADDRESS INCREMENT INHIBIT
622 033252 012777 010000 145770  MOV      #FMT22,@RHOF      ;FORMAT 16 BIT WORDS
623 033260 005077 145766  CLR      @RHCA      ;CYLINDER =0
624
625 ;SAVE REGISTERS
626
627 033264 004037 042152  JSR      R0,SAVER      ;SAVE
628 033270 001240          RHCS1      ;FROM
629 033272 002554          REINTO      ;TO
630 033274 000016          14.      ;NUMBER OF REGISTERS SAVED
631
632 ;NOW THE COMMAND IS GIVES TO
633 ;WRITE HEADER AND DATA FOR CYL=0, SECTOR=10
634 ;TRACK=0 IT COMES BACK AFTER ONE SECTOR
635 ;HAS PASSED
636
637 033276 012777 000001 145754  MOV      #DMD,@RHMR      ;SET DIAGNOSTIC MODE
638 033304 005277 145730  INC      @RHCS1      ;GO TO RHCS1 WITH 62
639 033310 012715 177672  MOV      #177672,@R5      ;TRY WRITING ALL BITS EXCEPT
640 ;GO, RMR, IE
641 033314 052737 000001 002574  BIS      #DMD,REINTO+20 ;SET DMD IN SAVED REGISTER RHMR
642 033322 052737 000004 002556  BIS      #RMR,REINTO+2  ;SET RMR IN SAVED REG. RHER1
643 033330 042737 000200 002576  BIC      #DRY,REINTO+22 ;CLEAR DRY IN RHDS1
644 033336 052737 040000 002576  BIS      #ERR,REINTO+22 ;SET ERR IN RHDS1
645 033344 052737 000001 002554  BIS      #GO,REINTO      ;SET GO IN SAVED REG. RHCS1
646 033352 042737 000200 002554  BIC      #RDY,REINTO      ;CLEAR RDY BIT
647
648 ;AFTER AN ATTEMPT TO WRITE INTO A REGISTER
649 ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
650
651 033360 004037 042152  JSR      R0,SAVER      ;SAVE
652 033364 001240          RHCS1      ;FROM
653 033366 001510          WRFROM      ;TO
654 033370 000016          14.      ;NUMBER
655
656 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
  
```

```
657 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
658 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
659 033372 113737 002573 001527 MOVB REINTO+17,WRFROM+17;SAVE UPPER RHAS
660
661
662 ;COMPARE REGISTERS BEFORE ATTEMPTED WRITE WITH AFTER
663
664 033400 004037 042354 JSR RO,COMPAR ;COMPAR
665 033404 002554 REINTO ;GO BUFFER
666 033406 001510 WRFROM ;TEST BUFFER
667 033410 000016 14. ;NUMBER
668 033412 033420 4$ ;RETURN FOR ERROR
669 033414 033420 4$ ;SAME
670 033416 033440 5$ ;RETURN FOR GOOD COMPARISON
671 033420 013705 046426 4$: MOV ERWORD,R5 ;GETTING READY TO INDEX
672 033424 060505 ADD R5,R5 ;DOUBLE ERROR WORD
673 033426 016537 001236 041130 MOV RHCS1-2(R5),REGADR ;FAILING REG. ADDRESS
674 033434 104001 EMT 1
675 033436 000207 RTS PC ;CHANGED WITH
676 ;AN ATTEMPT TO WRITE
677 ;DURING AN OPERATION
678 ;THE FOLLOWING CLEAR MAY SET THE ATA BIT BECAUSE GO IS HIGH
679
680 033440 004737 041460 5$: JSR PC,CLDISK ;CLEAR DISK
681
682
;*****
;*TEST 100 MAKE CURRENT CYLINDER = 1
;*****
033444 000004 TST100: SCOPE
033446 012706 001100 MOV #STACK,SP ;RESET STACK
033452 012737 000100 001432 MOV #100,TSTNM ;MOVE #100 TO TEST NUMBER
033460 004737 041460 JSR PC,CLDISK ;INIT DRIVE
033464 012777 000001 145566 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE
033472 004037 044032 JSR RO,MAKECYL ;DO SEEK COMMAND FOLLOWED BY AN INIT TO
033476 000001 .WORD 1 ;CHANGE RHCC TO CYLINDER 1
683
695
;*****
;*TEST 101 ERROR REG1 - BIT #7 - HEADER COMPARE ERROR
;*****
;*THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1
;*SECTOR=1, KEYS=1, 256 WORDS OF 177400
;*A READ HEADER AND DATA COMMAND IS GIVEN TO READ
;*CYLINDER=1, TRACK=1, SECTOR=1, KEY1=1, KEY2=1
;*REINTO BUFFER IS FILLED WITH 0
;*WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400
;*AFTER THE READ THE REINTO BUFFER IS EXPECTED TO
;*HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400
;*****
696 033500 000004 TST101: SCOPE
033502 012706 001100 MOV #STACK,SP ;RESET STACK
033506 012737 000101 001432 MOV #101,TSTNM ;MOVE #101 TO TEST NUMBER
033514 005037 046422 CLR NOSYNC ;SET FLAG SO THAT DATA SYNC
;AND DATA IS READ
;FILL SIMULATED DISK
033520 004737 043142 JSR PC,SETDSK ;SET UP SIMULATED DISK
```

:FILL REINTO BUFFER WITH 0

033524	004037	041376	JSR	RO,CLAREA	:FILL REINTO BUFFER
033530	002554		REINTO		:FROM LOCATION
033532	003554		REINTO+<256.*2>		:TO LOCATION
033534	000000		0		:DATA

:FILL WRFROM WITH 10000,401,1,1, AND ALL 177400

033536	012700	001510	MOV	#WRFROM,RO	
033542	012720	010000	MOV	#FMT22,(RO)+	:10000 INTO WRFROM
033546	012720	000401	MOV	#401,(RO)+	:401=TRACK1,SECTOR1
033552	012720	000001	MOV	#1,(RO)+	:1 INTO WRFROM+
033556	012720	000001	MOV	#1,(RO)+	:1 INTO WRFROM+6

:FILL ALL 0

033562	004037	041376	JSR	RO,CLAREA	:FILL WRFROM
033566	001520		WRFROM+10		:FROM
033570	002510		WRFROM+<256.*2>		:TO
033572	177400		177400		:DATA

:NOW GIVE A READ HEADER AND DATA COMMAND

:CYLINDER=1
:TRACK = 1
:SECTOR = 1

033574	004037	043270	JSR	RO,HCCRCE	
033600	000072		72		:READ HEADER AND DATA
033602	000001		1		:CYLINDER
033604	000001		1		:SECTOR
033606	000001		1		:TRACK
033610	177400		-256.		:WORD COUNT
033612	002554		REINTO		:RHBA BUFFER
033614	000000		0		:READ
033616	000001		1		:HEADER COMPARE
033620	000240		1\$:	NOP	:RETURN POINT FROM HCCRCE

697
698

:*****
:*TEST 102 MAKE CURRENT CYLINDER = 0
:*****

033622	000004		TST102:	SCOPE	
033624	012706	001100	MOV	#STACK,SP	:RESET STACK
033630	012737	000102	MOV	#102,TSTNM	:MOVE #102 TO TEST NUMBER
033636	004737	041460	JSR	PC,CLDISK	:INIT DRIVE
033642	012777	000001	MOV	#DMD,@RHMR	:SET DIAGNOSTIC MODE
033650	004037	044032	JSR	RO,MAKECYL	:DO SEEK COMMAND FOLLOWED BY AN INIT TO
033654	000000		WORD	0	:CHANGE RHCC TO CYLINDER 0

699
700

:*****
:*TEST 103 ERROR REG1 - BIT #7 - HEADER COMPARE ERROR

:*THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1
:*SECTOR=1, KEYS=1, 256 WORDS OF 177400


```

:*A READ HEADER AND DATA COMMAND IS GIVEN TO READ
:*CYLINDER=0, TRACK=0, SECTOR=1, KEY1=1, KEY2=1
:*REINTO BUFFER IS FILLED WITH 0
:*WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400
:*AFTER THE READ THE REINTO BUFFER IS EXPECTED TO
:*HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400
:*****

```

```

701 033656 000004
    033660 012706 001100
    033664 012737 000103 001432
    033672 005037 046422

```

```

TST103: SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #103,TSTNM     ;MOVE #103 TO TEST NUMBER
        CLR      NOSYNC        ;SET FLAG SO THAT DATA SYNC
                                ;AND DATA IS READ

```

;FILL SIMULATED DISK

```

033676 004737 043142          JSR      PC,SETDSK      ;SET UP SIMULATED DISK

```

;FILL REINTO BUFFER WITH 0

```

033702 004037 041376          JSR      R0,CLAREA     ;FILL REINTO BUFFER
033706 002554                                ;FROM LOCATION
033710 003554                                ;TO LOCATION
033712 000000                                ;DATA

```

;FILL WRFROM WITH 10000,401,1,1, AND ALL 177400

```

033714 012700 001510          MOV      #WRFROM,R0
033720 012720 010000          MOV      #FMT22,(R0)+ ;10000 INTO WRFROM
033724 012720 000401          MOV      #401,(R0)+  ;401=TRACK1,SECTOR1
033730 012720 000001          MOV      #1,(R0)+    ;1 INTO WRFROM+
033734 012720 000001          MOV      #1,(R0)+    ;1 INTO WRFROM+6

```

;FILL ALL 0

```

033740 004037 041376          JSR      R0,CLAREA     ;FILL WRFROM
033744 001520                                ;FROM
033746 002510                                ;TO
033750 177400                                ;DATA

```

;NOW GIVE A READ HEADER AND DATA COMMAND

```

;CYLINDER=0
;TRACK = 0
;SECTOR = 1

```

```

033752 004037 043270          JSR      R0,HCCRCE
033756 000072                72          ;READ HEADER AND DATA
033760 000000                0          ;CYLINDER
033762 000001                1          ;SECTOR
033764 000000                0          ;TRACK
033766 177400              -256.        ;WORD COUNT
033770 002554                REINTO     ;RHBA BUFFER
033772 000000                0          ;READ

033774 000001                1          ;HEADER COMPARE
033776 000240              1$: NOP      ;RETURN POINT FROM HCCRCE

```

702

703

034000 000004
034002 012706 001100
034006 012737 000104 001432
034014 004737 041460
034020 012777 000001 145232
034026 004037 044032
034032 000001

```
*****  
*TEST 104 MAKE CURRENT CYLINDER = 1  
*****  
TST104: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #104,TSTNM ;MOVE #104 TO TEST NUMBER  
JSR PC,CLDISK ;INIT DRIVE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
JSR RO,MAKECYL ;DO SEEK COMMAND FOLLOWED BY AN INIT TO  
.WORD 1 ;CHANGE RHCC TO CYLINDER 1
```

704
715

716 034034 000004
034036 012706 001100
034042 012737 000105 001432
034050 012737 177777 046422

```
*****  
*TEST 105 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR  
*THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1  
*SECTOR=1, KEYS=1, 256 WORDS OF 177400  
*A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=1  
*TRACK=1, SECTOR=1, KEY1=1, KEY2=1  
*WRFROM BUFFER IS FILLED WITH 125252  
*REINTO BUFFER IS FILLED WITH 177400  
*AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO  
*HAVE 177400  
*****
```

```
TST105: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #105,TSTNM ;MOVE #105 TO TEST NUMBER  
MOV #-1,NOSYNC ;SET FLAG SO THAT DATA SYNC  
;AND DATA IS NOT READ
```

;FILL SIMULATED DISK

034056 004737 043142

```
JSR PC,SETDSK ;SETUP SIMULATED DISK
```

;FILL WRFROM WITH 125252

034062 004037 041376
034066 001510
034070 002510
034072 125252

```
JSR RO,CLAREA ;FILL WRFROM BUFFER  
WRFROM ;FROM LOCATION  
WRFROM+<256.*2> ;TO LOCATION  
125252 ;DATA
```

;FILL REINTO WITH 256 WORDS OF 177400
;THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER
;AN ATTEMPT TO WRITE 125252

034074 004037 041376
034100 002554
034102 003554
034104 177400

```
JSR RO,CLAREA ;FILL REINTO BUFFER  
REINTO ;FROM LOCATION  
REINTO+<256.*2> ;TO  
177400
```

;NOW GIVE A WRITE DATA COMMAND
;CYLINDER = 1,
;TRACK = 1
;SECTOR = 1

034106 004037 043270
034112 000060
034114 000001

```
JSR RO,HCCRCE ;WRITE DATA  
60 ;CYLINDER  
1
```

034116	000001		1	:SECTOR
034120	000001		1	:TRACK
034122	177400		-256.	:WORD COUNT
034124	001510		WRFROM	:RHBA BUFFER
034126	000001		1	:WRITE
034130	000001		1	:HEADER COMPARE
034132	000240	1\$:	NOP	:RETURN POINT FROM HCCRCE

717
718

```
*****  
*TEST 106 MAKE CURRENT CYLINDER = 0  
*****  
TST106: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #106,TSTNM ;MOVE #106 TO TEST NUMBER  
JSR PC,CLDISK ;INIT DRIVE  
MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
JSR RO,MAKECYL ;DO SEEK COMMAND FOLLOWED BY AN INIT TO  
.WORD 0 ;CHANGE RHCC TO CYLINDER 0
```

719
720

```
*****  
*TEST 107 ERROR REG.1 - BIT #7 - HEADER COMPARE ERROR  
*THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1  
*SECTOR=1, KEYS=1, 256 WORDS OF 177400  
*A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=0  
*TRACK=0, SECTOR=1, KEY1=1, KEY2=1  
*WRFROM BUFFER IS FILLED WITH 125252  
*REINTO BUFFER IS FILLED WITH 177400  
*AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO  
*HAVE 177400  
*****
```

721	034170	000004			TST107: SCOPE
	034172	012706	001100		MOV #STACK,SP ;RESET STACK
	034176	012737	000107	001432	MOV #107,TSTNM ;MOVE #107 TO TEST NUMBER
	034204	012737	177777	046422	MOV #-1,NOSYNC ;SET FLAG SO THAT DATA SYNC ;AND DATA IS NOT READ

:FILL SIMULATED DISK

034212	004737	043142		JSR PC,SETDSK ;SETUP SIMULATED DISK
--------	--------	--------	--	-------------------------------------

:FILL WRFROM WITH 125252

034216	004037	041376		JSR RO,CLAREA ;FILL WRFROM BUFFER
034222	001510			WRFROM ;FROM LOCATION
034224	002510			WRFROM+<256.*2> ;TO LOCATION
034226	125252			125252 ;DATA

:FILL REINTO WITH 256 WORDS OF 177400
:THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER
:AN ATTEMPT TO WRITE 125252

034230	004037	041376		JSR RO,CLAREA ;FILL REINTO BUFFER
034234	002554			REINTO ;FROM LOCATION
034236	003554			REINTO+<256.*2> ;TO

034240 177400

177400

```

: NOW GIVE A WRITE DATA COMMAND
: CYLINDER = 0,
: TRACK = 0
: SECTOR = 1
  
```

034242 004037 043270
 034246 000060
 034250 000000
 034252 000001
 034254 000000
 034256 177400
 034260 001510
 034262 000001

```

JSR    RO,HCCRCE
60
0
1
0
-256.
WRFROM
1
  
```

```

: WRITE DATA
: CYLINDER
: SECTOR
: TRACK
: WORD COUNT
: RMB8 BUFFER
: WRITE
  
```

034264 000001
 034266 000240

```

1
1$:  NOP
: HEADER COMPARE
: RETURN POINT FROM HCCRCE
  
```

722
 723

```

:*****
: *TEST 110      RHER1 - BIT #8 - CRC ERROR (READING)
  
```

```

: *THE SIMULATED DISK IS SET TO READ CYLINDER=0, TRACK=1
: *SECTOR=1, KEYS=1, 256 WORDS OF 177400
: *A READ HEADER AND DATA COMMAND IS GIVEN TO READ
: *CYLINDER=0, TRACK=1, SECTOR=1, KEY1=1, KEY2=1
: *REINTO BUFFER IS FILLED WITH 0
: *WRFROM IS FILLED WITH 10000,401,1,1,1, AND ALL 177400
: *AFTER THE READ THE REINTO BUFFER IS EXPECTED TO
: *HAVE WHAT IS IN WRFROM - 10000,401,1,1 AND ALL 177400
  
```

724 034270 000004
 034272 012706 001100
 034276 012737 000110 001432
 034304 005037 046422

```

:*****
TST110: SCOPE
MOV    #STACK,SP      :RESET STACK
MOV    #110,TSTNM     :MOVE #110 TO TEST NUMBER
CLR    NOSYNC         :SET FLAG SO THAT DATA SYNC
:AND DATA IS READ
  
```

:FILL SIMULATED DISK

034310 004737 043142
 034314 005137 050206

```

JSR    PC,SETDSK      :SET UP SIMULATED DISK
COM    WCRC           :CHANCE CRC TO GIVE HCRC
  
```

:FILL REINTO BUFFER WITH 0

034320 004037 041376
 034324 002554
 034326 003554
 034330 000000

```

JSR    RO,CLAREA     :FILL REINTO BUFFER
REINTO :FROM LOCATION
REINTO+<256.*2>    :TO LOCATION
0       :DATA
  
```

:FILL WRFROM WITH 10000,401,1,1, AND ALL 177400

034332 012700 001510
 034336 012720 010000
 034342 012720 000401
 034346 012720 000001
 034352 012720 000001

```

MOV    #WRFROM,RO
MOV    #FMT22,(RO)+  :10000 INTO WRFROM
MOV    #401,(RO)+    :401=TRACK1,SECTOR1
MOV    #1,(RO)+      :1 INTO WRFROM+
MOV    #1,(RO)+      :1 INTO WRFROM+6
  
```

:FILL ALL 0

034356	004037	041376	JSR	RO,CLAREA	:FILL WRFROM
034362	001520		WRFROM+10		:FROM
034364	002510		WRFROM+<256.*2>		:TO
034366	177400		177400		:DATA

:NOW GIVE A READ HEADER AND DATA COMMAND
:CYLINDER=0
:TRACK = 1
:SECTOR = 1

034370	004037	043270	JSR	RO,HCCRCE	
034374	000072		72		:READ HEADER AND DATA
034376	000000		0		:CYLINDER
034400	000001		1		:SECTOR
034402	000001		1		:TRACK
034404	177400		-256.		:WORD COUNT
034406	002554		REINTO		:RHBA BUFFER
034410	000000		0		:READ
034412	000000		0		:CRC ERROR
034414	000240	1\$:	NOP		:RETURN POINT FROM HCCRCE

725
726

*TEST 111 RHER1 - BIT 8 - CRC ERROR (WRITING)
*THE SIMULATED DISK IS SET UP FOR CYLINDER=0, TRACK=1
*SECTOR=1, KEYS=1, 256 WORDS OF 177400
*A WRITE DATA COMMAND IS GIVEN TO WRITE CYLINDER=0
*TRACK=1, SECTOR=1, KEY1=1, KEY2=1
*WRFROM BUFFER IS FILLED WITH 125252
*REINTO BUFFER IS FILLED WITH 177400
*AFTER THE WRITE COMMAND THE DISK IS EXPECTED TO
*HAVE 177400

727	034416	000004				TST111: SCOPE
	034420	012706	001100			MOV #STACK,SP :RESET STACK
	034424	012737	000111	001432		MOV #111,TSTNM :MOVE #111 TO TEST NUMBER
	034432	012737	177777	046422		MOV #-1,NOSYNC :SET FLAG SO THAT DATA SYNC :AND DATA IS NOT READ

:FILL SIMULATED DISK

034440	004737	043142	JSR	PC,SETDSK	:SETUP SIMULATED DISK
034444	005137	050206	COM	WCRC	:CHANGE CRC TO GIVE HCRC

:FILL WRFROM WITH 125252

034450	004037	041376	JSR	RO,CLAREA	:FILL WRFROM BUFFER
034454	001510		WRFROM		:FROM LOCATION
034456	002510		WRFROM+<256.*2>		:TO LOCATION
034460	125252		125252		:DATA

:FILL REINTO WITH 256 WORDS OF 177400

:THIS IS WHAT IS EXPECTED TO BE ON DISK EVEN AFTER
:AN ATTEMPT TO WRITE 125252

034462	004037	041376	JSR	RO,CLAREA	:FILL REINTO BUFFER
034466	002554		REINTO		:FROM LOCATION
034470	003554		REINTO+<256.*2>		:TO
034472	177400		177400		

:NOW GIVE A WRITE DATA COMMAND
:CYLINDER = 0,
:TRACK = 1
:SECTOR = 1

034474	004037	043270	JSR	RO,HCCRCE	
034500	000060		60		:WRITE DATA
034502	000000		0		:CYLINDER
034504	000001		1		:SECTOR
034506	000001		1		:TRACK
034510	177400		-256.		:WORD COUNT
034512	001510		WRFROM		:RHBA BUFFER
034514	000001		1		:WRITE
034516	000000		0		:CRC ERROR
034520	000240	1\$:	NOP		:RETURN POINT FROM HCCRCE

728
729
730
731
732
733
734
735
736
737

:SET UP FOR THE TWO LAST SECTOR TRANSFERRED TESTS FOLLOWING

034522	005737	001434	TST	RP06	:MOVE RP06 FLAG TO ITSELF TO TEST
034526	001401		BEQ	2\$:IF = 0 TREAT DRIVE AS RP04
034530	000402		BR	3\$:TREAT AS RP06 - DO NEXT 'MAKECL' & TEST
034532	000137	035206	JMP	DOG	:DO SECOND FOLLOWING 'MAKECL' AND TEST

2\$:
3\$:

::*****
:*TEST 112 MAKE CURRENT CYLINDER = 814.
:*****

034536	000004		TST112: SCOPE		
034540	012706	001100	MOV	#STACK,SP	:RESET STACK
034544	012737	000112	MOV	#112,TSTNM	:MOVE #112 TO TEST NUMBER
034552	004737	041460	JSR	PC,CLDISK	:INIT DRIVE
034556	012777	000001	MOV	#DMD,@RHMR	:SET DIAGNOSTIC MODE
034564	004037	044032	JSR	RO,MAKECYL	:DO SEEK COMMAND FOLLOWED BY AN INIT TO
034570	001456		.WORD	814.	:CHANGE RHCC TO CYLINDER 814.

738
746

::*****
:*TEST 113 RHDS1 (BIT #10) - LAST SECTOR TRANSFERRED, 'LST'
:*WRITE CYLINDER 814. FORMAT 16 BITS PER WORD
:*TRACK 18., SECTOR 21., KEYS 0, NUMBER OF WORDS
:*256. OF 377
:*'LST' BIT # 10 - RHDS1, SHOULD SET AFTER WRITE
:*IS COMPLETE.
:*****

034572	000004		TST113: SCOPE		
034574	012706	001100	MOV	#STACK,SP	:RESET STACK
034600	012737	000113	MOV	#113,TSTNM	:MOVE #113 TO TEST NUMBER

```
034606 004037 041376 JSR R0,CLAREA ;CLEAR SIMULATED DISK
034612 050224 .WORD DISK ;FROM
034614 051250 .WORD TOLGAP+16 ;TO
034616 000000 .WORD 0 ;DATA
```

;THESE ARE SETUP FOR DISKLESS USE ONLY

```
034620 012737 011456 046306 MOV #814.,!FMT22,CYL ;CYLINDER 814.
034626 112737 000022 046311 MOV #18.,SECOTR+1 ;16 BITS PER WORD
034634 112737 000025 046310 MOV #21.,SECOTR ;TRACK 18.
034642 005037 046312 CLR KEY1 ;SECTOR 21.
034646 005037 046314 CLR KEY2 ;KEY1 0
034652 012737 000400 046354 MOV #256.,NOWORD ;KEY2 0
034660 012737 000001 046316 MOV #1,X ;NO OF DATA WORDS
034666 004537 042666 JSR R5,CRC ;WRITE DATA
034672 046306 CYL ;GO TO CALCULATE CRC
034674 050206 WCRC
```

;THESE ARE REGULAR SETUPS

```
034676 004037 041376 JSR R0,CLAREA ;FILL WRITE BUFFER WITH 377
034702 001510 WRFROM ;FROM LOCATION
034704 002510 WRFROM+<256.*2> ;TO LOCATION
034706 000377 377 ;DATA
034710 004737 041460 JSR PC,CLDISK ;SETUP GENERAL REGISTERS
034714 012777 177400 144310 MOV #-256.,@RHWC ;256. DATA WORDS
034722 012777 001510 144304 MOV #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
034730 012746 000025 MOV #21.,-(SP) ;SECTOR 21.
034734 112766 000022 000001 MOV #18.,1(SP) ;TRACK 18.
034742 012677 144276 MOV (SP)+,@RHDST ;SECTOR 21. TRACK 18.
034746 012777 010000 144274 MOV #FMT22,@RHOF ;16 BITS PER WORD FORMAT
034754 012777 001456 144270 MOV #814.,@RHCA ;CYLINDER 814.
034762 004737 041514 JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
034766 104401 062726 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
034772 000000 HALT ;STOP THE TEST
034774 013711 001462 MOV WRIDAT,@R1 ;WRITE DATA=60
035000 005037 001406 CLR ERFLG$ ;CLEAR ERROR FLAG
035004 004737 046146 JSR PC,COMHD ;WRITE DATA
```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 ;FROM THE 'COMHD' ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
 ;HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
 ;AND SYNCs WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.

```
035010 004737 041064 JSR PC,PUTREG ;SAVE REGISTERS
035014 005737 001406 TST ERFLG$ ;HAVE ANY ERRORS OCCURED?
035020 001062 BNE 5$ ;BRANCH IF YES
035022 012700 000377 MOV #377,R0 ;GOOD DATA
035026 012701 050224 MOV #DISK,R1 ;DATA WRITTEN INTO 'DISK'
035032 012702 000400 MOV #256.,R2 ;COUNTER
035036 012737 000401 046426 1$: MOV #256.+1,ERWORD ;FOR ERROR WORD
035044 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
035046 001424 BEQ 3$ ;BRANCH IF GOOD
035050 010037 001124 MOV R0,$GDDAT ;GOOD DATA
```

```
035054 014137 001126      MOV      -(R1), $BDDAT      :BAD DATA
035060 160237 046426      SUB      R2, ERWORD        :ERROR WORD NO
035064 005737 001406      TST      ERFLG$           :ANY ERRORS ALREADY THERE?
035070 001002      BNE      2$                :BRANCH IF YES
035072 104004      EMT      4                  :
035074 000401      BR       1064$             :BRANCH TO AVOID PRINTING NEXT ERROR

035076      2$:
035076 104005      EMT      5
035100 005721      1064$: TST      (R1)+         :UNDO -(R1) FOR BAD DATA
035102 017746 144032      MOV      @SWR, -(SP)       :GET SWITCH SETTING
035106 042716 177177      BIC      #177177, (SP)     :KEEP ONLY SWITCH 7 AND 8
035112 022726 000200      CMP      #SW07, (SP)+     :IS 7 SET AND 8 RESET
035116 001402      BEQ     4$                :BRANCH OUT IF YES
035120 005302      3$:   DEC      R2          :IF NOT COUNT 256 WORDS
035122 001345      BNE     1$                :BRANCH IF 256. NCT DONE

748
749 035124 013746 001336      4$:   MOV      DS1, -(SP)     :GET RHDS1
750 035130 042716 001000      BIC      #PROG, (SP)      :CLEAR PROG
751 035134 022726 002700      CMP      #LST!DPR!DRY!VV, (SP)+ :IS 'LST' HIGH ?
752 035140 001412      BEQ     5$                :BRANCH IF GOOD
753 035142 013737 001262 041130      MOV      RHDS1, REGADR :FAILING REG. ADDRESS
754 035150 012737 002700 001124      MOV      #LST!DPR!DRY!VV, $GDDAT :GOOD DATA
755 035156 013737 001336 001126      MOV      DS1, $BDDAT     :BAD DATA
756 035164 104001      EMT      1
757
758
759
760
761 035166 013737 001240 035176      5$:   MOV      RHCS1, 6$
762 035174 104415      WAT
763 035176 000000      6$:   0
764 035200 000200      RDY
765
766 035202 000137 035652      JMP      CAT              :DON'T DO THE RP04 'LST' TEST FOLLOWING
767
768 035206
```

```
DOG:
*****
*TEST 114      MAKE CURRENT CYLINDER = 410.
*****
```

```
035206 000004      TST114: SCOPE
035210 012706 001100      MOV      #STACK, SP      :RESET STACK
035214 012737 000114 001432      MOV      #114, TSTNM     :MOVE #114 TO TEST NUMBER
035222 004737 041460      JSR      PC, CLDISK      :INIT DRIVE
035226 012777 000001 144024      MOV      #DMD, @RHMR     :SET DIAGNOSTIC MODE
035234 004037 044032      JSR      RO, MAKECYL     :DO SEEK COMMAND FOLLOWED BY AN INIT TO
035240 000632      .WORD   410.           :CHANGE RHCC TO CYLINDER 410.
```

```
769
777
*****
*TEST 115      RHDS1 (BIT #10) - LAST SECTOR TRANSFERRED, 'LST'
*WRITE CYLINDER 410.  FORMAT 16 BITS PER WORD
*TRACK 18.  SECTOR 21.  KEYS 0.  NUMBER OF WORDS
*256.  OF 377
*'LST' BIT #10 - RHDS1 SHOULD SET AFTER THE
*WRITE IS COMPLETED
*****
```



```

778 035242 000004          TST115: SCOPE
035244 012706 001100      MOV      #STACK,SP      ;RESET STACK
035250 012737 000115 001432  MOV      #115,TSTNM     ;MOVE #115 TO TEST NUMBER

035256 004037 041376      JSR      R0,CLAREA      ;CLEAR SIMULATED DISK
035262 050224              .WORD   DISK           ;FROM
035264 051250              .WORD   TOLGAP+16      ;TO
035266 000000              .WORD   0              ;DATA
  
```

;THESE ARE SETUP FOR DISKLESS USE ONLY

```

035270 012737 010632 046306      MOV      #410,!FMT22,CYL ;CYLINDER 410.
035276 112737 000022 046311      MOVB    #18.,SECOTR+1   ;16 BITS PER WORD
035304 112737 000025 046310      MOVB    #21.,SECOTR    ;TRACK 18.
035312 005037 046312              CLR     KEY1           ;SECTOR 21.
035316 005037 046314              CLR     KEY2           ;KEY1 0
035322 012737 000400 046354      MOV     #256.,NOWORD   ;KEY2 0
035330 012737 000001 046316      MOV     #1,X           ;NO OF DATA WORDS
035336 004537 042666      JSR     R5,CRC         ;WRITE DATA
035342 046306              CYL                    ;GO TO CALCULATE CRC
035344 050206              WCRC
  
```

;THESE ARE REGULAR SETUPS

```

035346 004037 041376      JSR     R0,CLAREA      ;FILL WRITE BUFFER WITH 377
035352 001510      WRFROM ;FROM LOCATION
035354 002510      WRFROM+<256.*2>      ;TO LOCATION
035356 000377      377                  ;DATA
035360 004737 041460      JSR     PC,CLDISK     ;SETUP GENERAL REGISTERS
035364 012777 177400 143640      MOV     #-256.,@RHWC   ;256. DATA WORDS
035372 012777 001510 143634      MOV     #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
035400 012746 000025              MOV     #21.,-(SP)    ;SECTOR 21.
035404 112766 000022 000001      MOVB    #18.,1(SP)    ;TRACK 18.
035412 012677 143626              MOV     (SP)+,@RHDST  ;SECTOR 21. TRACK 18.
035416 012777 010000 143624      MOV     #FMT22,@RHOF  ;16 BITS PER WORD FORMAT
035424 012777 000632 143620      MOV     #410.,@RHCA   ;CYLINDER 410.
035432 004737 041514      JSR     PC,CHECKT     ;CHECK THAT DVA,RDY,DPR,DRY = 1
035436 104401 062726      TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
                                ;TINUE TESTING IF BOTH AREN'T TRUE
035442 000000      HALT                ;STOP THE TEST
035444 013711 001462      MOV     WRIDAT,@R1    ;WRITE DATA=60
035450 005037 001406      CLR     ERFLG$        ;CLEAR ERROR FLAG
035454 004737 046146      JSR     PC,COMHD      ;WRITE DATA
  
```

;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 ;FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
 ;HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
 ;AND SYNCs WERE CORRECTLY DETECTED, DATA IS TO BE CHECKED.

```

035460 004737 041064      JSR     PC,PUTREG     ;SAVE REGISTERS
035464 005737 001406      TST     ERFLG$        ;HAVE ANY ERRORS OCCURED?
035470 001062      BNE     5$           ;BRANCH IF YES
035472 012700 000377      MOV     #377,R0       ;GOOD DATA
035476 012701 050224      MOV     #DISK,R1      ;DATA WRITTEN INTO 'DISK'
035502 012702 000400      MOV     #256.,R2     ;COUNTER
  
```

```

035506 012737 000401 046426 1$: MOV #256,+1,ERWORD ;FOR ERROR WORD
035514 020021 CMP R0,(R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
035516 001424 BEQ 3$ ;BRANCH IF GOOD
035520 010037 001124 MOV R0,$GDDAT ;GOOD DATA
035524 014137 001126 MOV -(R1),$BDDAT ;BAD DATA
035530 160237 046426 SUB R2,ERWORD ;ERROR WORD NO
035534 005737 001406 TST ERFLG$ ;ANY ERRORS ALREADY THERE?
035540 001002 BNE 2$ ;BRANCH IF YES
035542 104004 EMT 4
035544 000401 BR 1064$ ;BRANCH TO AVOID PRINTING NEXT ERROR
  
```

```

035546 2$: EMT 5
035546 104005 1064$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
035550 005721 MOV @SWR,-(SP) ;GET SWITCH SETTING
035552 017746 143362 BIC #177177,(SP) ;KEEP ONLY SWITCH 7 AND 8
035556 042716 177177 CMP #SW07,(SP)+ ;IS 7 SET AND 8 RESET
035562 022726 000200 BEQ 4$ ;BRANCH OUT IF YES
035566 001402 3$: DEC R2 ;IF NOT COUNT 256 WORDS
035570 005302 BNE 1$ ;BRANCH IF 256. NOT DONE
035572 001345
  
```

```

779
780 035574 013746 001336 4$: MOV DS1,-(SP) ;GET RHDS1
781 035600 042716 001000 BIC #PROG,(SP) ;CLEAR PROG BIT
782 035604 022726 002700 CMP #LST!DPR!DRY!VV,(SP)+ ;IS 'LST' HIGH ?
783 035610 001412 BEQ 5$ ;WAIT FOR 'RDY' IF GOOD
784 035612 013737 001262 041130 MOV RHDS1,REGADR ;FAILING REG. ADDRESS
785 035620 012737 002700 001124 MOV #LST!DPR!DRY!VV,$GDDAT ;GOOD DATA
786 035626 013737 001336 001126 MOV DS1,$BDDAT ;BAD DATA
787 035634 104001 EMT 1
788 ;LAST SECTOR ON LAST TRACK ON LAST
789 ;CYLINDER WAS WRITTEN - 'VV' BIT #6
790 ;MAY OR MAY NOT BE HIGH
791
  
```

```

792 035636 013737 001240 035646 5$: MOV RHCS1,6$ ;SET UP 'WAT' SUBROUTINE
793 035644 104415 WAT
794 035646 000000 6$: 0 ;RHCS1 ADDRESS
795 035650 000200 RDY ;WAIT FOR 'RDY' BIT
796
812
  
```

```

*****
;*TEST 116 ERROR REGISTER 1 - BIT #9 AOE
;*A WRITE DATA COMMAND IS GIVEN TO CYLINDER 410./814.
;*SECTOR 21 TRACK 18, KEYS 0, DATA 377
;*WORD COUNT REGISTER FOR 522 (256+66+200) WORDS
;*
;*AFTER 256 WORDS HAVE BEEN WRITTEN
;* 'AOE' SHOULD COME UP
  
```

```

;*RHWC WILL SHOW 200 BECAUSE THE SILO IS 66 WORDS AND
;*256 WORDS HAVE BEEN WRITTEN - TOTAL 322
;*THIS IS 200 SHORT OF 522
*****
  
```

```

035652
035652
813 035652 012706 001100 MOV #STACK,SP ;RESET STACK
814 035656 012737 000116 001432 MOV #116,TSTNM ;MOVE #116 TO TEST NUMBER
815 035664 004737 041460 JSR PC,CLDISK ;INIT AND SET UP GENERAL REG. CORRES.
  
```

```

816 035670 004037 041376      JSR    R0,CLAREA      ;CLEAR SIMULATED DISK
817 035674 050224              .WORD  DISK           ;FROM
818 035676 051250              .WORD  TOLGAP+16     ;TO
819 035700 000000              .WORD  0              ;DATA
820
821                          ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
822                          ;AND WILL HANDLE RP04 OR RP06 DRIVES
823
824 035702 005737 001434      TST    RP06           ;MOVE RP06 FLAG TO ITSELF TO TEST
825 035706 001404      BEQ    10$           ;TREAT DRIVE AS RP04 IF = 0
826
827 035710 012737 011456 046306  MOV    #814.,!FMT22,CYL;CYLINDER 814., 16 BITS PER WORD
828 035716 000403      BR     11$           ;TREAT DRIVE AS RP06
829
830 035720 012737 010632 046306 10$:  MOV    #410.,!FMT22,CYL;CYLINDER 410., 16 BITS PER WORD
831                          ;TREAT DRIVE AS RP04
832
833 035726 112737 000022 046311 11$:  MOVB   #18.,SECOTR+1   ;TRACK 18.
834 035734 112737 000025 046310  MOVB   #21.,SECOTR    ;SECTOR 21.
835 035742 005037 046312      CLR    KEY1           ;KEY1 0
836 035746 005037 046314      CLR    KEY2           ;KEY2 0
837 035752 012737 000400 046354  MOV    #256.,NOWORD   ;NO OF DATA WORDS
838 035760 012737 000001 046316  MOV    #1,X           ;WRITE DATA
839 035766 004537 042666      JSR    R5,CRC        ;GO TO CALCULATE CRC
840 035772 046306      CYL
841 035774 050206      WCRC
842
843                          ;THESE ARE REGULAR SETUPS
844
845 035776 004037 041376      JSR    R0,CLAREA     ;FILL WRITE BUFFER WITH 377
846 036002 001510      WRFROM              ;FROM
847 036004 002510      WRFROM+<256.*2>    ;TO
848 036006 000377      377                ;DATA
849 036010 004737 041460      JSR    PC,CLDISK    ;SETUP GENERAL REGISTERS
850 036014 012777 176766 143210  MOV    #-522.,@RHWC  ;522. DATA WORDS
851 036022 012777 001510 143204  MOV    #WRFROM,@RHBA ;STARTING ADDRESS OF WRITE BUFFER
852 036030 012746 000025      MOV    #21.,-(SP)   ;SECTOR 21.
853 036034 112766 000022 000001  MOVB   #18.,1(SP)   ;TRACK 18.
854 036042 012677 143176  MOV    (SP)+,@RH DST ;SECTOR 21. TRACK 18.
855 036046 012777 010000 143174  MOV    #FMT22,@RHOF ;16 BITS PER WORD FORMAT
856
857                          ;CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
858                          ;AND LOAD CYLINDER ADDRESS REGISTER WITH THE PROPER NUMBER
859
860 036054 005737 001434      TST    RP06           ;MOVE FLAG TO ITSELF TO TEST
861 036060 001404      BEQ    12$           ;TREAT AS RP04 IF = 0
862 036062 012777 001456 143162  MOV    #814.,@RHCA   ;CYLINDER 814.
863 036070 000403      BR     13$           ;TREAT AS RP06
864 036072 012777 000632 143152 12$:  MOV    #410.,@RHCA   ;CYLINDER 410.
865 036100      13$:
866
867 036100 004737 041514      JSR    PC,CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
868 036104 104401 062726  TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
                        ;TINUE TESTING IF BOTH AREN'T TRUE
869 036110 000000      HALT                ;STOP THE TEST
868 036112 013711 001462  MOV    WRIDAT,@R1    ;WRITE DATA=60
869 036116 005037 001406  CLR    ERFLG$       ;CLEAR ERROR FLAG
  
```

```

870
871 ;THE REGISTERS WILL BE SAVED IN REINTO BUFFER
872
873 036122 004037 042152 JSR RO,SAVER ;SAVE
874 036126 001232 RHWC ;FROM
875 036130 002554 REINTO ;TO
876 036132 000023 19. ;NUMBER SAVED
877
878 ;GIVE WRITE DATA COMMAND
879
880 036134 004737 046146 JSR PC,COMHD ;WRITE DATA COMMAND
881
882 ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
883
884 036140 005737 001436 TST RH70 ;CHECK FOR RH70 CONTROLLER
885 036144 001407 BEQ JP1 ;SKIP RH70 CODE AND DO RH11 IF NOT
886
887 036146 012737 177376 002554 VAR1: MOV #-258.,REINTO ;SAVED RHWC SHOULD BE = 258.
888 036154 012737 002530 002556 VAR2: MOV #WRFROM+<2*256.>+<2*8.>,REINTO+2
889 ;SAVED RHBA SHOULD BE WRFROM+256+8
890 036162 000406 BR JP2 ;SKIP NEXT RH11 CODE
891
892 036164 012737 177470 002554 JP1: MOV #-200.,REINTO ;SAVED RHWC SHOULD BE = 200.
893 036172 012737 002714 002556 MOV #WRFROM+<2*256.>+<2*66.>,REINTO+2
894 ;SAVED RHBA SHOULD BE WRFROM+256+66
895
896 036200 052737 000200 002560 JP2: BIS #OR,REINTO+4 ;SAVED RHCS2
897 036206 042737 000100 002560 BIC #IR,REINTO+4 ;SAVED RHCS2
898 036214 052737 140000 002562 BIS #SC!TRE,REINTO+6;SAVED RHCS1 SHOULD HAVE 'SC' & 'TRE'
899 036222 012737 001000 002564 MOV #AOE,REINTO+10 ;SAVED RHER1 SHOULD HAVE 'AOE'
900 036230 017737 143010 002566 MOV @RHDST,REINTO+12;SAVED RHDST SHOULD HAVE=
901 ;RHDST IS UNDEFINED
902
903 ;CHECK TO SEE WHAT TYPE OF DRIVE IS BEING TESTED
904 ;AND SET UP CYLINDER ADDRESS ACCORDINGLY
905
906 036236 005737 001434 TST RP06 ;MOVE RP06 FLAG TO ITSELF TO TEST
907 036242 001404 BEQ 14$ ;TREAT AS RP04 IF = 0
908 036244 012737 001457 002574 MOV #815.,REINTO+20;SAVED DESIRED CYLINDER ADDRESS
909 036252 000403 BR 15$ ;TREAT AS RP06
910 036254 012737 000633 002574 14$: MOV #411.,REINTO+20;SAVED DESIRED CYLINDER ADDRESS
911
912 036262 013737 001416 002600 15$: MOV ATTENT,REINTO+24 ;SAVED RHAS SHOULD HAVE APPRO. BIT
913 036270 052737 000001 002602 BIS #DMD,REINTO+26;SAVED RHM
914 036276 052737 142000 002604 BIS #ATA!ERR!LST,REINTO+30 ;SAVED RHDS1
915
916 ;AFTER A WRITE DATA COMMAND WITH 'AOE' ERROR
917 ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
918
919 036304 004037 042152 JSR RO,SAVER ;SAVE
920 036310 001232 RHWC ;FROM
921 036312 001510 WRFROM ;TO
922 036314 000021 17. ;NUMBER OF REGISTERS SAVED
923
924 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
925 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
926 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE

```

```

927
928 036316 113737 002601 001535      MOVB    REINTO+25,WRFROM+25;SAVE UPPER RHAS
929
930      ;COMPARE REGISTERS BEFORE WRITE DATA COMMAND
931      ;WITH AFTER COMMAND
932
933 036324 004037 042354      JSR     R0,COMPARE      ;COMPARE
934 036330 002554      REINTO      ;GOOD BUFFER
935 036332 001510      WRFROM      ;TEST BUFFER
936 036334 000021      17.        ;NUMBER OF REGISTERS
937 036336 036344      1$         ;RETURN FOR ERROR
938 036340 036344      1$         ;SAME
939 036342 036364      2$         ;RETURN FOR GOOD COMPARISON
940
941 036344 013705 046426      1$:      MOV     ERWORD,R5      ;GETTING READY TO INDEX
942 036350 060505      ADD      R5,R5          ;DOUBLE ERROR WORD
943 036352 016537 001230 041130  MOV     RHWC-2(R5),REGADR ;FAILING REG. ADDRESS
944 036360 104001      EMT      1
945      ;REGISTER CHANGE
946 036362 000207      RTS     PC              ;RETURN FOR FURTHER COMPARISONS
947      ;NO ERRORS
948 036364 005037 001406      2$:      CLR     ERFLG$      ;CLEAR ERROR FLAG
949
950
951      ;DATA IS TO BE CHECKED HERE
952
953 036370 004737 041064      JSR     PC,PUTREG      ;SAVE REGISTERS
954 036374 012700 000377      MOV     #377,R0        ;GOOD DATA
955 036400 012701 050224      MOV     #DISK,R1       ;DATA WRITTEN INTO 'DISK'
956 036404 012702 000400      MOV     #256.,R2       ;COUNTER
957 036410 012737 000400 046426 3$:      MOV     #256.,ERWORD   ;FOR ERROR WORD
958 036416 020021      CMP     R0,(R1)+       ;COMPARE GOOD DATA WITH DATA ON DISK
959 036420 001424      BEQ     6$            ;BRANCH IF GOOD
960 036422 010037 001124      MOV     R0,$GDDAT     ;GOOD DATA
961 036426 014137 001126      MOV     -(R1),$BDDAT  ;BAD DATA
962 036432 160237 046426      SUB     R2,ERWORD     ;ERROR WORD NO
963 036436 005737 001406      TST     ERFLG$      ;ANY ERRORS ALREADY THERE?
964 036442 001002      BNE     4$            ;BRANCH IF YES
965 036444 104004      EMT
966 036446 000401      BR      5$            ;BRANCH TO AVOID PRINTING NEXT ERROR
967 036450      4$:
968 036450 104005      EMT      5
969 036452 005721      5$:      TST     (R1)+         ;UNDO -(R1) FOR BAD DATA
970 036454 017746 142460      MOV     @SWR,-(SP)     ;GET SWITCH SETTING
971 036460 042716 177177      BIC     #177177,(SP)  ;KEEP ONLY SWITCH 7 AND 8
972 036464 022726 000200      CMP     #SW07,(SP)+   ;IS 7 SET AND 8 RESET
973 036470 001402      BEQ     7$            ;BRANCH OUT IF YES
974 036472 005302      6$:      DEC     R2            ;IF NOT COUNT 256 WORDS
975 036474 001345      BNE     3$            ;BRANCH IF 256. NOT DONE
976
977 036476      7$:
978      ;*****
      ;*TEST 117      MAKE CURRENT CYLINDER = 0
      ;*****
      TST117: SCOPE
      MOV     #STACK,SP      ;RESET STACK
  
```

```

036504 012737 000117 001432      MOV    #117,TSTNM      ;MOVE #117 TO TEST NUMBER
036512 004737 041460              JSR    PC,CLDISK      ;INIT DRIVE
036516 012777 000001 142534      MOV    #DMD,@RHMR     ;SET DIAGNOSTIC MODE
036524 004037 044032              JSR    R0,MAKECYL     ;DO SEEK COMMAND FOLLOWED BY AN INIT TO
036530 000000                      .WORD  0              ;CHANGE RHCC TO CYLINDER 0

979
987
:*****
:*TEST 120      ERROR REGISTER 1 - BIT #10 'IAE'
:*A READ HEADER AND DATA IS GIVEN TO TRACK 20, SECTOR 0
:*
:*AN INDEX PULSE IS GIVEN TO GET RHLA TO 0
:*
:*IAE BIT SHOULD SET
:*****
TST120: SCOPE
988 036532 000004              MOV    #STACK,SP      ;RESET STACK
989 036534 012706 001100      MOV    #120,TSTNM     ;MOVE #120 TO TEST NUMBER
990 036540 012737 000120 001432  JSR    PC,CLDISK      ;CLEAR REGISTERS AND SET UNIT NO.
991
992
993 036552 012777 000001 142500  ;GIVE INDEX PULSE
994 036560 052777 000004 142472  MOV    #DMD,@RHMR     ;SET DIAGNOSTIC MODE
995 036566 042777 000004 142464  BIS    #MINX,@RHMR    ;SET INDEX
996
997
998
999
1000 036574 012777 177400 142430  MOV    #-256,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
1001 036602 012700 002554      MOV    #REINTO,R0     ;THESE TWO INSTRUCTIONS GETS
1002 036606 010077 142422      MOV    R0,@RHBA       ;ADDR, OF WRFROM INTO R0 AND
1003
1004 036612 012720 010000      MOV    #FMT22,(R0)+   ;BUS ADDRESS REGISTER
1005
1006 036616 012720 012000      MOV    #12000,(R0)+  ;FORMAT=16 BIT WORDS
1007 036622 005020              CLR    (R0)+          ;CYLINDER=0
1008 036624 005020              CLR    (R0)+          ;TRACK=20 SECTOR=0 KEYS=0
1009 036626 012705 000400      MOV    #256,R5       ;KEY1=0
1010 036632 012720 177777      1$: MOV    #-1,(R0)+   ;KEY2=0
1011 036636 005305              DFC    R5             ;COUNTER
1012 036640 001374              BNE    1$            ;MOVE ALL ONES FOR DATA
1013 036642 012777 012000 142374  MOV    #12000,@RHDST ;BRANCH IF DATA NOT COMPLETE
1014
1015 036650 004737 041514              JSR    PC,CHECKT     ;TRACK=20 SECTOR=0
036654 104401 062726              TYPE  ,CPHALT        ;CHECK THAT DVA,RDY,DPR,DRY = 1
036660 000000              HALT                 ;AND THAT NO OTHERS = 1. CANNOT CON-
;TINUE TESTING IF BOTH AREN'T TRUE
;STOP THE TEST

1016
1017 036662 013711 001470      MOV    REFOR,@R1      ;GET READY FOR WRITE HEADER AND
1018
1019 036666 005037 001406              CLR    ERFLG$        ;DATA WITH 62 IN RHCS1
1020 036672 012777 010000 142350  MOV    #FMT22,@RHOF   ;CLEAR ERROR FLAG
1021 036700 005077 142346      CLR    @RHCA          ;FORMAT BIT=1 (16 BIT WORDS)
1022
1023
1024 036704 004037 042152      ;THE REGISTERS WILL BE SAVED IN REINTO BUFFER
1025 036710 001232              JSR    R0,SAVER      ;SAVE
1026 036712 002554              RHC   REINTO         ;FROM
;TO

```

```

1027 036714 000023          19.          ;NUMBER SAVED
1028
1029          ;GO TO WRITE HEADER AND DATA
1030
1031 036716 013700 001260      MOV      RHMR,R0 ;NOW R0 WAS MAINTENANCE REG. ADDR.
1032 036722 012710 000001      MOV      #DMD,@R0 ;SET DIAGNOSTIC MODE
1033 036726 052777 000001 142304  BIS      #GO,@RHCS1 ;GO
1034
1035          ;CHANGE SAVED REGISTERS TO EXPECTED VALUE
1036 036734 052737 140000 002562  BIS      #SC!TRE,REINTO+6 ;SAVED RHCS1
1037 036742 012737 002000 002564  MOV      #IAE,REINTO+10 ;SAVED RHER1
1038 036750 012737 012001 002566  MOV      #12001,REINTO+12 ;SAVED RHDST
1039 036756 013737 001416 002600  MOV      ATTENT,REINTO+24 ;SAVED RHAS
1040 036764 052737 000001 002602  BIS      #DMD,REINTO+26 ;SAVED RHMR
1041 036772 052737 140000 002604  BIS      #ATA!ERR,REINTO+30 ;SAVED RHDS1
1042
1043          ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
1044 037000 004037 042152      JSR      R0,SAVER ;SAVE
1045 037004 001232              RHC      ;FROM
1046 037006 001510              WRFROM ;TO
1047 037010 000023              19.      ;NUMBER OF REGISTERS SAVED
1048
1049          ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1050          ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1051          ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1052 037012 113737 002601 001535  MOVB     REINTO+25,WRFROM+25;SAVE UPPER RHAS
1053
1054          ;COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
1055          ;WITH AFTER COMMAND
1056
1057 037020 004037 042354      JSR      R0,COMPAR ;COMPARE
1058 037024 002554              REINTO ;GOOD BUFFER
1059 037026 001510              WRFROM ;TEST BUFFER
1060 037030 000021              17.    ;NUMBER OF REGISTERS
1061 037032 037040              2$    ;RETURN FOR ERROR
1062 037034 037040              2$    ;SAME
1063 037036 037060              3$    ;RETURN FOR GOOD COMPARISON
1064
1065 037040 013705 046426 2$:   MOV      ERWORD,R5 ;GETTING READY TO INDEX
1066 037044 040505              ADD      R5,R5 ;DOUBLE ERROR WORD
1067 037046 016537 001230 041130  MOV      RHWC-2(R5),REGADR ;FAILING REG. ADDRESS
1068 037054 104001              EMT      1
1069
1070 037056 000207              RTS      PC ;REGISTER CHANGE
1071
1072
1073
1074 037060 004737 041460 3$:   JSR      PC,CLDISK ;CLEAR GO BIT
1075
1086
*****
;*TEST 121 ERROR REGISTER 1- BIT #10 'IAE'
;*A WRITE HEADER AND DATA IS GIVEN TO SECTOR 22
;*TRACK 0 CYLINDER 0
;*
;*WORD COUNT IS SET TO 256.
;*
;*AN INDEX PULSE IS GIVEN TO GET RHLA TO 0

```

```

: *
: * IAE BIT SHOULD SET
: *****
TST121: SCOPE
1087 037064 000004          MOV    #STACK,SP      ;RESET STACK
1088 037066 012706 001100  MOV    #121,TSTNM     ;MOVE #121 TO TEST NUMBER
1089 037100 004737 041460  JSR    PC,CLDISK     ;CLEAR REGISTERS AND SET UNIT NO.
1090
1091                      ;GIVE INDEX PULSE
1092 037104 012777 000001 142146  MOV    #DMD,@RHMR     ;SET DIAGNOSTIC MODE
1093 037112 052777 000004 142140  BIS    #MINX,@RHMR    ;SET INDEX
1094 037120 042777 000004 142132  BIC    #MINX,@RHMR    ;CLEAR INDEX
1095
1096                      ;THESE ARE REGULAR SETUPS
1097
1098 037126 012777 177400 142076  MOV    #-256,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
1099 037134 012700 001510          MOV    #WRFROM,R0     ;THESE TWO INSTRUCTIONS GETS
1100 037140 010077 142070          MOV    R0,@RHBA      ;ADDR. OF WRFROM INTO R0 AND
1101                      ;BUS ADDRESS REGISTER
1102 037144 012720 010000          MOV    #FMT22,(R0)+  ;FORMAT=16 BIT WORDS
1103                      ;CYLINDER=0
1104 037150 012720 000026          MOV    #22,(R0)+     ;TRACK=0, SECTOR=22, KEYS=0
1105 037154 005020          CLR    (R0)+         ;KEY1=0
1106 037156 005020          CLR    (R0)+         ;KEY2=0
1107 037160 012705 000400          MOV    #256,R5       ;COUNTER
1108 037164 012720 177777 1$: MOV    #-1,(R0)+     ;MOVE ALL ONES FOR DATA
1109 037170 005305          DEC    R5
1110 037172 001374          BNE    1$
1111 037174 012777 000026 142042  MOV    #22,@RHDST    ;BRANCH IF DATA NOT COMPLETE
1112                      ;TRACK=0 SECTOR=22
1113 037202 004737 041514          JSR    PC,CHECKT     ;CHECK THAT DVA,RDY,DPR,DRY = 1
1114 037206 104401 062726          TYPE    ,CPHALT     ;AND THAT NO OTHERS = 1. CANNOT CON-
1115                      ;TINUE TESTING IF BOTH AREN'T TRUE
1116                      ;STOP THE TEST
1117 037212 000000          HALT
1118
1119 037214 013711 001464          MOV    WRIFOR,@R1    ;GET READY FOR WRITE HEADER AND
1120                      ;DATA WITH 62 IN RHCS1
1121 037220 005037 001406          CLR    ERFLG$ ;CLEAR ERROR FLAG
1122 037224 012777 010000 142016  MOV    #FMT22,@RHOF  ;FORMA BIT=1 (16 BIT WORDS)
1123 037232 005077 142014          CLR    @RHCA        ;CYLINDER =0
1124
1125                      ;AS EXCEPTION IS ASSERTED BEFORE RUN IS
1126                      ;LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED
1127                      ;BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS
1128                      ;ON DIFFERENT UNITS
1129
1130                      ;THE REGISTERS WILL BE SAVED IN REINTO BUFFER
1131 037236 004037 042152          JSR    R0,SAVER     ;SAVE
1132                      ;FROM
1133 037242 001242          RHER1              ;TO
1134 037244 002554          REINTO             ;NUMBER SAVED
1135 037246 000015          13.
1136
1137                      ;GO TO WRITE HEADER AND DATA
1138
1139 037250 013700 001260          MOV    RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADDR.
1140 037254 012710 000001          MOV    #DMD,@R0     ;SET DIAGNOSTIC MODE
1141 037260 052777 000001 141752  BIS    #GO,@RHCS1   ;GO
  
```



```

1137
1138 ;CHANGE SAVED REGISTERS TO EXPECTED VALUE
1139 037266 012737 002000 002554 MOV #IAE,REINTO ;SAVED RHER1
1140 037274 012737 000027 002556 MOV #23,REINTO+2;SAVED RHDST
1141 037302 013737 001416 002570 MOV ATTENT,REINTO+14 ;SAVED RHAS
1142 037310 052737 000001 002572 BIS #DMD,REINTO+16 ;SAVED RHMR
1143 037316 052737 140000 002574 BIS #ATA!ERR,REINTO+20 ;SAVED RHDS1
1144
1145 ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
1146 037324 004037 042152 JSR RO,SAVER ;SAVE
1147 037330 001242 RHER1 ;FROM
1148 037332 001510 WRFROM ;TO
1149 037334 000015 13. ;NUMBER OF REGISTERS SAVED
1150
1151 ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1152 ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1153 ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1154 037336 113737 002571 001525 MOVB REINTO+15,WRFROM+15;SAVE UPPER RHAS
1155
1156 ;COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
1157 ;WITH AFTER COMMAND
1158 JSR RO,COMPAR ;COMPARE
1159 037344 004037 042354 REINTO ;GOOD BUFFER
1160 037350 002554 WRFROM ;TEST BUFFER
1161 037352 001510 13. ;NUMBER OF REGISTERS
1162 037354 000015 2$ ;RETURN FOR ERROR
1163 037356 037364 2$ ;SAME
1164 037360 037364 3$ ;RETURN FOR GOOD COMPARISON
1165 037362 037404
1166
1167 037364 013705 046426 2$: MOV ERWORD,R5 ;GETTING READY TO INDEX
1168 037370 060505 ADD R5,R5 ;DOUBLE ERROR WORD
1169 037372 016537 001240 041130 MOV RHER1-2(R5),REGADR ;FAILING REG. ADDRESS
1170 037400 104001 EMT 1
1171 ;REGISTER CHANGE
1172 037402 000207 RTS PC ;RETURN FOR FURTHER COMPARISONS
1173
1174 ;NO ERRORS
1175
1176 037404 004737 041460 3$: JSR PC,CLDISK ;CLEAR GO BIT
1177
1187
;*****
;*TEST 122 ERROR REGISTER 1- BIT #10 'IAE'
;*A WRITE DATA IS GIVEN TO SECTOR 0
;*TRACK 0 CYLINDER 411
;*
;*WORD COUNT IS SET TO 256.
;*AN INDEX PULSE IS GIVEN TO GET RHLA TO 0
;*
;*IAE BIT SHOULD SET
;*****
TST122: SCOPE
1188 037410 000004 MOV #STACK,SP ;RESET STACK
1189 037412 012706 001100 MOV #122,TSTNM ;MOVE #122 TO TEST NUMBER
1190 037416 012737 000122 001432 JSR PC,CLDISK ;CLEAR REGISTERS AND SET UNIT NO.
1191 037424 004737 041460
1192 ;GIVE INDEX PULSE
  
```

```

1193 037430 012777 000001 141622      MOV    #DMD,@RHMR      ;SET DIAGNOSTIC MODE
1194 037436 052777 000004 141614      BIS    #MINX,@RHMR    ;SET INDEX
1195 037444 042777 000004 141606      BIC    #MINX,@RHMR    ;CLEAR INDEX
1196
1197      ;THESE ARE REGULAR SETUPS
1198
1199 037452 012777 177400 141552      MOV    #-256.,@RHWC   ;256 DATA WORDS 4 HEADER WORDS
1200 037460 012700 001510                MOV    #WRFROM,RO     ;THESE TWO INSTRUCTIONS GETS
1201 037464 010077 141544                MOV    RO,@RHBA       ;ADDR. OF WRFROM INTO RO AND
1202                                ;BUS ADDRESS REGISTER
1203 037470 012705 000400                MOV    #256.,R5       ;COUNTER
1204 037474 012720 177777      1$:   MOV    #-1,(RO)+     ;MOVE ALL ONES FOR DATA
1205 037500 005305                DEC    R5
1206 037502 001374                BNE    1$             ;BRANCH IF DATA NOT COMPLETE
1207 037504 012777 000000 141532      MOV    #0.,@RHDST     ;TRACK=0 SECTOR=0
1208
1209 037512 004737 041514                JSR    PC,CHECKT      ;CHECK THAT DVA,RDY,DPR,DRY = 1
      037516 104401 062726                TYPE    ,CPHALT      ;AND THAT NO OTHERS = 1. CANNOT CON-
      ;TINUE TESTING IF BOTH AREN'T TRUE
      037522 000000                HALT                  ;STOP THE TEST
1210
1211 037524 013711 001462                MOV    WRIDAT,@R1     ;GET READY FOR WRITE
1212                                ;DATA WITH 60 IN RHCS1
1213 037530 005037 001406                CLR    ERFLG$        ;CLEAR ERROR FLAG
1214 037534 012777 010000 141506      MOV    #FMT22,@RHOF   ;FORMA BIT=1 (16 BIT WORDS)
1215
1216 037542 005737 001434                TST    RP06          ;MOVE FLAG TO ITSELF TO TEST
1217 037546 001404                BEQ    4$            ;TREAT DRIVE AS RP04 IF FLAG = 0
1218
1219 037550 012777 001457 141474      MOV    #815.,@RHCA    ;CYLINDER = 815 (ONE TOO MANY)
1220 037556 000403                BR     5$            ;TREAT DRIVE AS RP06
1221
1222 037560 012777 000633 141464      4$:   MOV    #411.,@RHCA   ;CYLINDER = 411 (ONE TOO MANY)
1223                                ;TREAT DRIVE AS RP04
1224
1225      ;AS EXCEPTION IS ASSERTED BEFORE RUN IS
1226      ;LATCHED RHWC,RHBA,RHCS1,RHCS2 CANNOT BE CHECKED
1227      ;BECAUSE RHWC WILL VARY DEPENDING UPON GATE DELAYS
1228      ;ON DIFFERENT UNITS
1229
1230      ;THE REGISTERS WILL BE SAVED IN REINTO BUFFER
1231 037566 004037 042152      5$:   JSR    RO,SAVER      ;SAVE
1232 037572 001242                RHER1                ;FROM
1233 037574 002554                REINTO                ;TO
1234 037576 000015                13.                  ;NUMBER SAVED
1235
1236      ;GO TO WRITE HEADER AND DATA
1237
1238 037600 013700 001260                MOV    RHMR,RO        ;NOW RO HAS MAINTENANCE REG. ADDR.
1239 037604 012710 000001                MOV    #DMD,@RO      ;SET DIAGNOSTIC MODE
1240 037610 052777 000001 141422      BIS    #GO,@RHCS1    ;GO
1241
1242      ;CHANGE SAVED REGISTERS TO EXPECTED VALUE
1243 037616 012737 002000 002554      MOV    #IAE,REINTO   ;SAVED RHER1
1244 037624 012737 000001 002556      MOV    #1.,REINTO+2 ;SAVED RHDST
1245 037632 013737 00 416 002570      MOV    ATTENT,REINTO+14 ;SAVED RHAS
1246 037640 052737 000001 002572      BIS    #DMD,REINTO+16 ;SAVED RHMR
  
```

```
1247 037646 052737 140000 002574      BIS      #ATA!ERR,REINTO+20 ;SAVED RHD$1
1248
1249      ;SAVE REGISTERS AGAIN SO THAT COMPARES CAN BE DONE
1250 037654 004037 042152      JSR      R0,SAVER      ;SAVE
1251 037660 001242      RHER1      ;FROM
1252 037662 001510      WRFROM      ;TO
1253 037664 000015      13.      ;NUMBER OF REGISTERS SAVED
1254
1255      ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
1256      ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
1257      ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
1258 037666 113737 002571 001525      MOVB     REINTO+15,WRFROM+15;SAVE UPPER RHAS
1259
1260
1261      ;COMPARE REGISTERS BEFORE READ IN PRESET COMMAND
1262      ;WITH AFTER COMMAND
1263 037674 004037 042354      JSR      R0,COMPAR      ;COMPARE
1264 037700 002554      REINTO      ;GOOD BUFFER
1265 037702 001510      WRFROM      ;TEST BUFFER
1266 037704 000015      13.      ;NUMBER OF REGISTERS
1267 037706 037714      2$      ;RETURN FOR ERROR
1268 037710 037714      2$      ;SAME
1269 037712 037734      3$      ;RETURN FOR GOOD COMPARISON
1270
1271 037714 013705 046426      2$:     MOV     ERWORD,R5      ;GETTING READY TO INDEX
1272 037720 060505      ADD     R5,R5      ;DOUBLE ERROR WORD
1273 037722 016537 001240 041130      MOV     RHER1-2(R0),REGADR ;FAILING REG. ADDRESS
1274 037730 104001      EMT     1
1275
1276 037732 000207      RTS     PC      ;REGISTER CHANGE
1277      ;RETURN FOR FURTHER COMPARISONS
1278
1279      ;NO ERRORS
1280 037734 004737 041460      3$:     JSR     PC,CLDISK      ;CLEAR GO BIT
1281
1282
1283      ;*****
1284      ;*TEST 123      END OF DRIVE
1285      ;*THIS IS THE END OF TEST FOR ONE DRIVE
1286      ;*IF THERE ARE MORE DRIVES THEN THE PROGRAM
1287      ;*JUMPS TO TEST 5 FOR NEXT DRIVE TEST
1288      ;*END PASS IS REACHED ONLY AFTER ALL DRIVES ARE COMPLETE
1289      ;******
1290      ;TST123: SCOPE
1291      MOV     #1,$TIMES      ;;DO 1 ITERATION
1292      MOV     #0,PS      ;REINSTATE PS TO 0
1293      TYPE   ,65$      ;;TYPE ASCIZ STRING
1294      BR     64$      ;;GET OVER THE ASCIZ
1295      ;:65$: .ASCIZ <CRLF>/TOTAL ERRORS FOR THIS PASS ON UNIT NO. /
1296      64$:
1297      MOV     UNIT,-(SP)      ;GET READY TO TYPE UNIT NUMBER
1298      TYPDS
1299      TYPE   ,67$      ;;TYPE ASCIZ STRING
1300      BR     66$      ;;GET OVER THE ASCIZ
1301      ;:67$: .ASCIZ /= /
1302      66$:
1303      MOV     $ERTTL,-(SP)      ;GET READY TO TYPE NUMBER OF ERRORS
1304      TYPDS
```

1296	040064	005037	001112		CLR	\$ERTTL		:CLEAR TOTAL NUMBER OF ERRORS
1297	040070	005037	001102		CLR	\$TSTNM		:CLEAR TEST NUMBER
1298	040074	005737	001402		TST	SELECT		:STARTING FROM 210 ?
1299								
1300	040100	001413			BEQ	3\$:TEST NEXT DRIVE IF NOT
1301								:CONTINUE ON THIS ONE IF SO
1302	040102	005237	001100		INC	\$PASS		:INCREASE PASS COUNT
1303	040106	104401	040271		TYPE	\$ENDMG		:TYPE END PASS #
1304	040112	013746	001100		MOV	\$PASS,-(SP)		
1305	040116	104405			TYPDS			
1306	040120	104401	040266		TYPE	\$ENULL		
1307	040124	000137	006440		JMP	TSTS		:DO NEXT TESTS
1308								
1309	040130	005337	001376	3\$:	DEC	NOUNITS		:NO. OF UNITS PRESENT DECREMENTED
1310	040134	001413			BEQ	\$EOP		:BRANCH IF ALL DRIVES COMPLETE
1311	040136	013700	001374		MOV	UNIT,R0		:UNIT UNDER TEST
1312	040142	012701	001354		MOV	#UNITS,R1		:TABLE
1313	040146	022100		1\$:	CMP	(R1)+,R0		:IS THIS UNIT JUST TESTED
1314	040150	001401			BEQ	2\$:BRANCH IF YES
1315	040152	000775			BR	1\$:BRANCH IF NO
1316	040154	011137	001374	2\$:	MOV	(R1),UNIT		:THIS IS NEXT UNIT
1317	040160	000137	006440		JMP	TSTS		:GO FOR NEXT TESTS

1
 2
 3

.SBTTL SUBROUTINES

.SBTTL END OF PASS ROUTINE

::*****
 :*INCREMENT THE PASS NUMBER (\$PASS)
 :*TYPE 'END PASS #XXXXX' (WHERE X IS A DECIMAL NUMBER)
 :*IF THERES A MONITOR GO TO IT
 :*IF THERE ISN'T JUMP TO TST1

040164				\$EOP:	SCOPE		
040164	000004				CLR	\$STNM	::ZERO THE TEST NUMBER
040166	005037	001102			CLR	\$TIMES	::ZERO THE NUMBER OF ITERATIONS
040172	005037	001212			INC	\$PASS	::INCREMENT THE PASS NUMBER
040176	005237	001100			BIC	#100000,\$PASS	::DON'T ALLOW A NEG. NUMBER
040202	042737	100000	001100		DEC	(PC)+	::LOOP?
040210	005327			\$EOPCT:	.WORD	1	
040212	000001				BGT	\$DOAGN	::YES
040214	003022				MOV	(PC)+,@(PC)+	::RESTORE COUNTER
040216	012737			\$ENDCT:	.WORD	1	
040220	000001				\$EOPCT		
040222	040212				TYPE	,\$SENDMG	::TYPE 'END PASS #'
040224	104401	040271			MOV	\$PASS,-(SP)	::SAVE \$PASS FOR TYPEOUT
040230	013746	001100			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
040234	104405				TYPE	,\$NULL	::TYPE A NULL CHARACTER
040236	104401	040266		\$GET42:	MOV	@#42,R0	::GET MONITOR ADDRESS
040242	013700	000042			BEQ	\$DOAGN	::BRANCH IF NO MONITOR
040246	001405				RESET		::CLEAR THE WORLD
040250	000005			\$ENDAD:	JSR	PC,(R0)	::GO TO MONITOR
040252	004710				NOP		::SAVE ROOM
040254	000240				NOP		::FOR
040256	000240				NOP		::ACT11
040260	000240			\$DOAGN:	JMP	@(PC)+	::RETURN
040262	000137				\$RTNAD:	.WORD	TST1
040264	005052				\$ENULL:	.BYTE	-1,-1,0
040266	377	377	000		\$SENDMG:	.ASCIZ	<15><12>/END PASS #/
040271	015	012	105				

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
040774

```

:*****
:HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
:ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
:PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

:WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
:THE PROGRAM GOES BACK TO CAN BE CHANGED.
:THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
:1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
:2. LOOP ON ERROR SWITCH MUST BE SET
:3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
:IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
:THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
:TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
:THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
:COMES TO THE END OF THE TEST UNDER CONSIDERATION.

:AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
:NORMAL OPERATION WILL CONTINUE.
:*****
  
```

```

TESTAD: 0 ;FIRST ADDRESS OF TEST

OPERSEL:
CLR PS ;MAKE PROCESSOR STATUS ZERO
TYPE ,65$ ;:TYPE ASCIZ STRING
BR 64$ ;:GET OVER THE ASCIZ
::65$: .ASCIZ <CRLF>/THE PROGRAM WAS IN TEST NUMBER /
64$:
MOV TSTNM,-(SP) ;GET READY TO TYPE TEST
TYPOC ;NUMBER
TYPE ,67$ ;:TYPE ASCIZ STRING
BR 66$ ;:GET OVER THE ASCIZ
::67$: .ASCIZ <CRLF>/THE LOOP BACK PC WAS /
66$:
MOV $LPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE ,69$ ;:TYPE ASCIZ STRING
BR 68$ ;:GET OVER THE ASCIZ
::69$: .ASCIZ <CRLF>/SET SWITCH FOR LOOP ON ERROR OR LOOP ON TEST./
68$:
TYPE ,71$ ;:TYPE ASCIZ STRING
BR 70$ ;:GET OVER THE ASCIZ
::71$: .ASCIZ <CRLF>/TYPE THE FIRST PC OF THE TEST TO BE LOOPEd ON,/
70$:
TYPE ,73$ ;:TYPE ASCIZ STRING
BR 72$ ;:GET OVER THE ASCIZ
::73$: .ASCIZ <CRLF>/FOLLOWED BY A CARRIAGE RETURN /<CRLF>
72$:
RDOCT
ADD #2,(SP) ;GET LPADR
MOV (SP)+,$LPADR
TYPE ,75$ ;:TYPE ASCIZ STRING
BR 74$ ;:GET OVER THE ASCIZ
::75$: .ASCIZ <CRLF>/TYPE THE PC WHERE YOU WANT THE PROGRAM TO LOOP BACK/
74$:
  
```

40	040774	104401	041002	TYPE	77\$::TYPE ASCIZ STRING
	041000	000423		BR	76\$::GET OVER THE ASCIZ
	041050			::77\$:	.ASCIZ	<CRLF>/TO, FOLLOWED BY A CARRIAGE RETURN /<CRLF>
	76\$:					
41	041050	104412		RDOCT		
42	041052	012637	001110	MOV	(SP)+,\$LPERR	:GET LPERR
43	041056	013746	001106	MOV	\$LPADR,-(SP)	
44	041062	000002		RTI		

SAVE REGISTERS ROUTINE

```

1          .SBTTL SAVE REGISTERS ROUTINE
2
3
4          ;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
5          ;IN MEMORY LOCATIONS TAGED FROM 'WC' TO 'EC2'
6          ;
7          ;
8          ;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
9          ;AND NOT THE REGISTERS THEMSELVES. THIS WILL MAKE
10         ;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
11
12 041064   010046   PUTREG:  MOV    R0,-(SP)      ;;PUSH R0 ON STACK
13 041064   010146   MOV    R1,-(SP)      ;;PUSH R1 ON STACK
14 041066   010246   MOV    R2,-(SP)      ;;PUSH R2 ON STACK
15 041070   012700   001232  MOV    #RHWC,R0      ;;STARTING ADDRESS OF REG
16 041072   012701   001306  MOV    #WC,R1        ;;STARTING ADDRESS OF WERE SAVED
17 041076   012702   000023  MOV    #RHCC-RHWC+2/2,R2 ;;NUMBER OF REG. INTO R2
18 041102   013021   10$:    MOV    @R0+,(R1)+   ;;SAVE HARDWARE REG.
19 041106   013021   DEC    R2
20 041110   005302   BNE   10$
21 041112   001375   MOV    (SP)+,R2     ;;POP STACK INTO R2
22 041114   012602   MOV    (SP)+,R1     ;;POP STACK INTO R1
23 041116   012601   MOV    (SP)+,R0     ;;POP STACK INTO R0
24 041120   012600
25 041122   000207   RTS    PC
26
27         .SBTTL FLOAT 1 AND 0
28
29         ;FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
30         ;ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4
31
32 MASK:    0          ;BITS UNDER TEST
33 LERR:    0          ;ERROR HLT ADDRESS
34 REGADR:  0
35
36 BITST:  MOV    (R5)+,MASK  ;FETCH DATA MASK
37          MOV    (R5)+,R4   ;GET ADDRESS OF REG. UNDER TEST
38          MOV    R4,REGADR
39          MOV    R5,LERR    ;GET ERROR RETURN ADDR.
40          ADD    #4,R5      ;MODIFY RETURN ADDR. TO JUMP OVER RTS
41          MOV    #1,R3     ;INITIALIZE DATA PATTERN
42
43 BLT1:    JSR    PC,BLT2    ;OUTPUT FLOATING ZERO
44          JSR    PC,BLT2    ;OUTPUT FLOATING ONE
45          CLC
46          ROL    R3        ;SHIFT PATTERN
47          TST    R3
48          BNE   BLT1      ;BRANCH IF NOT COMPLETE
49          RTS    R5        ;RETURN TO TEST
50
51 BLT2:    COM    R3        ;COMPLEMENT PATTERN
52          MOV    #BLT3,LAD  ;SET SCOPE LOOP
53          BIT    #SW09,@SWR ;LOOP ON ERROR
54          BEQ   4$        ;BRANCH IF NO
55          TSTB  SERFLG     ;ANY ERRORS
56          BEQ   4$        ;BRANCH IF NO
57          RESET
58          MOV    UNIT,@RHCS2 ;START WITH AN INIT
59          JSR    PC,$TKINT  ;SET UNIT NUMBER UNDER TEST
60          ;INITILIZE TK

```



```

53
54 041244 010337 001124      4$:  MOV    R3,$GDDAT      ;INIT FOR SCOPING LOOPS
55 041250 005137 041124      COM    MASK            ;STORE GOOD DATA
56 041254 043737 041124 001124  BIC    MASK,$GDDAT    ;AND MASK WITH PATTERN
57 041262 005137 041124      COM    MASK            ;CLEAR THE REST
58 041266 013714 001124      MOV    $GDDAT,(R4)    ;RESTORE MASK
59 041272 011437 001126      MOV    (R4),$BDDAT   ;OUTPUT TO REGISTER
60 041276 005137 041124      COM    MASK            ;INPUT FROM REGISTER
61 041302 043737 041124 001126  BIC    MASK,$BDDAT    ;AND MASK OUT RECEIVED DATA
62 041310 005137 041124      COM    MASK            ;RESTORE MASK
63 041314 023737 001124 001126  CMP    $GDDAT,$BDDAT ;IS DATA CORRECT
64 041322 001424              BEQ    1$              ;BRANCH IF GOOD
65 041324 011437 001126      MOV    (R4),$BDDAT
66 041330 023704 001240      CMP    RHCS1,R4       ;REGISTER UNDER TEST RHCS1?
67 041334 0C1004              BNE    2$              ;BRANCH IF NOT
68 041336 052737 004200 001124  BIS    #RDY!DVA,$GDDAT;SET RDY AND DVA
69 041344 000410              BR     3$
70 041346 023704 001236      2$:  CMP    RHCS2,R4       ;REGISTER UNDER TEST RHCS2?
71 041352 001005              BNE    3$              ;BRANCH IF NOT
72 041354 011446              MOV    @R4,-(SP)      ;GET RHCS2
73 041356 042716 177477      BIC    #^C<IR!OR>,(SP);KEEP IR AND OR BIT
74 041362 052637 001124      BIS    (SP)+,$GDDAT  ;SET IR OR BITS IF NEEDED
75 041366 004777 177534      3$:  JSR    PC,@LERR      ;GO TO REPORT ERROR
76 041372 000240              NOP
77 041374 000207      1$:  RTS    PC           ;REPLACE BY 104420 FOR LOCAL SCOPE LOOP
    
```

```

1      .SBTTL CLEAR MEMORY ROUTINE
2
3      ;THIS CLEARS ANY BLOCK OF MEMORY
4      ;FILLING IT WITH ANY DATA
5
6      ;CALL
7      JSR      R0,CLAREA
8      .WORD    X           ;STARTING ADDRESS OF BLOCK
9      .WORD    Y
10     .WORD    Z           ;DATA TO BE FILLED
11
12     ;R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
13     ;R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
14     ;R3 WILL HAVE DATA TO BE FILLED
15     ;TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
16
17     041376
18     041376 010146
19     041400 010246
20     041402 010346
21     041404 012001
22     041406 012002
23     041410 012003
24     041412 160102
25     041414 062702 000002
26     041420 010321
27     041422 005302
28     041424 005302
29     041426 001374
30     041430 012603
31     041432 012602
32     041434 012601
33     041436 000200
    
```

```

CLAREA:
    MOV      R1,-(SP)      ;;PUSH R1 ON STACK
    MOV      R2,-(SP)      ;;PUSH R2 ON STACK
    MOV      R3,-(SP)      ;;PUSH R3 ON STACK
    MOV      (R0)+,R1      ;FROM
    MOV      (R0)+,R2      ;TO
    MOV      (R0)+,R3      ;DATA
    SUB      R1,R2         ;NO. OF LOCATIONS MINUS TWO
    ADD      #2,R2         ;GET TWICE NO OF LOCATIONS
1$:  MOV      R3,(R1)+     ;MOVE IN DATA
    DEC      R2
    DEC      R2
    BNE     1$            ;BRANCH IF NOT COMPLETE
    MOV      (SP)+,R3     ;;POP STACK INTO R3
    MOV      (SP)+,R2     ;;POP STACK INTO R2
    MOV      (SP)+,R1     ;;POP STACK INTO R1
    RTS      R0           ;RETURN
    
```

```
1          .SBTTL LOCAL TRAPS
2
3 041440 000000          LAD: 0
4
5 041442 032777 001000 137470 T.SCOPI: BIT #SW09,@SWR
6 041450 001402          BEQ 1$
7 041452 013716 041440          MOV LAD,(SP)
8 041456 000002          1$: RTI
9
10         ;EXAMPLE OF THE USE OF THE ABOVE
11         ;THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO 'NEWTST'
12         ;CALL
13         ;:      MOV      #X,LAD
14         ;X:      ---      ---
15         ;:      ---      ---
16         ;:      ---      ---
17         ;:      SCOP1
18
19         .SBTTL CLEAR DISK ROUTINE
20
21 041460 013701 001240          CLDISK: MOV      RHCS1,R1          ;R1 WILL BE CONTROL AND STATUS1
22 041464 013702 001236          MOV      RHCS2,R2          ;R2 WILL BE CONTROL AND STATUS2
23 041470 013703 001262          MOV      RHDS1,R3          ;R3 WILL BE DISK STATUS REGISTER1
24 041474 013704 001242          MOV      RHER1,R4          ;R4 WILL BE ERROR REGISTER #1
25
26 041500 012712 000040          MOV      #CLR,@R2          ;CLEAR ALL REG.
27 041504 013712 001374          MOV      UNIT,@R2          ;REINSTATE UNIT NO.
28 041510 005011          CLR      @R1          ;CLEAR FUNCTION BITS
29 041512 000207          RTS      PC
```

```

1      .SBTTL  CHECK DISK STATUS ROUTINES
2
3      ;THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
4      ;AND CHECKS THAT DEVICE PRESENT (DPR)  DEVICE READY (DRY) IN RHDS1 = 1
5      ;IT ALSO CHECKS THAT NO OTHER BITS IN THESE REGISTERS = 1
6
7 041514 011637 001414      CHECKT: MOV      (SP),PCJSR      ;SAVE PC OF JSR+4
8 041520 162737 000004 001414  SUB      #4,PCJSR      ;GET PC OF JSR
9 041526 004737 041064      JSR      PC,PUTREG     ;SAVE REGISTERS
10 041532 022737 004200 001314  CMP      #DVA!RDY,CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
11                                     ;AND BE READY
12 041540 001423      BEQ      3$          ;BRANCH IF GOOD
13
14 041542 032737 004000 001314  BIT      #DVA,CS1     ;BAD SO TEST DEVICE AVAILABLE
15 041550 001004      BNE      1$          ;BRANCH IF DVA THERE
16 041552 010137 001122      MOV      R1,$BDADR    ;ADDRESS OF BAD REGISTER (RHCS1)
17 041556 104026      EMT      26
18                                     ;AVAILABLE AT START OF TEST
19 041560 000413      BR       3$          ;BRANCH TO NEXT COMPARE
20 041562 032737 000200 001314  1$: BIT      #RDY,CS1   ;TEST READY
21 041570 001003      BNE      2$          ;IF RDY THERE BRANCH
22 041572 010137 001122      MOV      R1,$BDADR    ;ADDRESS OF BAD REGISTER (RHCS1)
23 041576 104026      EMT      26
24                                     ;RIGHT AT START OF TEST
25 041600 000403      BR       3$          ;BRANCH TO NEXT COMPARE
26 041602 010137 001122      MOV      R1,$BDADR    ;ADDRESS OF BAD REGISTER (RHCS1)
27 041606 104026      EMT      26
28                                     ;THAN DVA AND RDY SET
29                                     ;ALL OTHER BITS SHOULD BE 0
30
31 041610 013746 001336      3$: MOV      DS1,-(SP)   ;GET RHDS1
32 041614 042716 001100      BIC      #VV!PROG,(SP) ;CLEAR VV AND PROGRAMABLE BIT
33 041620 022726 000600      CMP      #DPR!DRY,(SP)+;RHDS1 SHOULD HAVE THESE SET
34 041624 001424      BEQ      8$          ;BRANCH IF THEY ARE
35
36 041626 032737 000400 001336  4$: BIT      #DPR,DS1   ;TEST DRIVE PRESENT
37 041634 001004      BNE      5$          ;CONTINUE IF THERE
38 041636 010337 001122      MOV      R3,$BDADR    ;ADDRESS OF BAD REGISTER (RHDS1)
39 041642 104026      EMT      26
40 041644 000413      BR       7$          ;BRANCH OUT
41 041646 032737 000200 001336  5$: BIT      #DRY,DS1   ;TEST DRIVE READY
42 041654 001004      BNE      6$          ;IF DPR WAS THERE, BRANCH IF GOOD
43 041656 010337 001122      MOV      R3,$BDADR    ;ADDRESS OF BAD REGISTER (RHDS1)
44 041662 104026      EMT      26
45 041664 000403      BR       7$          ;BRANCH OUT
46 041666 010337 001122      MOV      R3,$BDADR    ;ADDRESS OF BAD REGISTER (RHDS1)
47 041672 104026      EMT      26
48                                     ;THAN MOL, DRY, DPR, SET
49                                     ;ALL OTHER BITS SHOULD BE 0
50 041674 000207      7$: RTS      PC      ;RETURN TO TEST AND HALT
51
52 041676 062716 000006      8$: ADD      #6,(SP)   ;ADJUST STACK TO GET OVER HALT IN TEST
53 041702 000207      RTS      PC      ;RETURN TO TEST AND CONTINUE TESTING
    
```

```

1
2
3
4
5 041704 011637 001414
6 041710 162737 000004 001414
7 041716 004737 041064
8 041722 032737 000200 001314
9
10 041730 001004
11 041732 010137 001122
12 041736 104026
13
14 041740 000427
15 041742 032737 004000 001314 1$:
16
17 041750 001004
18 041752 010137 001122
19 041756 104026
20
21 041760 000417
22 041762 032737 000200 001336 2$:
23 041770 001004
24 041772 010337 001122
25 041776 104026
26 042000 000407
27 042002 032737 000400 001336 3$:
28 042010 001004
29 042012 010337 001122
30 042016 104026
31
32 042020 000207 4$:
33
34 042022 062716 000006 5$:
35 042026 000207

;THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
;AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1

CHECKE: MOV (SP),PCJSR ;SAVE PC OF JSR+4
SUB #4,PCJSR ;GET PC OF JSR
JSR PC,PUTREG ;READ & SAVE REGISTERS
BIT #RDY,CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
;AND BE READY
BNE 1$ ;BRANCH IF GOOD
MOV R1,$BDADR ;FAILING REGISTER
EMT 26

;DOES NOT HAVE DVA, RDY
BR 4$ ;BRANCH OUT
BIT #DVA,CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
;AND BE READY
BNE 2$ ;BRANCH IF GOOD
MOV R1,$BDADR ;FAILING REGISTER
EMT 26

;DOES NOT HAVE DVA, RDY
BR 4$ ;BRANCH OUT
BIT #DRY,DS1 ;RHDS1 SHOULD HAVE DPR, DRY
BNE 3$ ;BRANCH IF THERE
MOV R3,$BDADR ;FAILING REGISTER RHDS1
EMT 26

;BRANCH OUT
BR 4$ ;RHDS1 SHOULD HAVE DPR, DRY
BIT #DPR,DS1 ;BRANCH OUT AND CONTINUE IF THERE
BNE 5$ ;FAILING REGISTER RHDS1
MOV R3,$BDADR
EMT 26

RTS PC ;RETURN TO TEST AND HALT

ADD #6,(SP) ;ADJUST STACK TO GET OVER HALT IN TEST
RTS PC ;RETURN TO TEST AND CONTINUE TESTING
    
```

```

1      .SBTTL WAIT LOOP
2
3      ;ONE LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
4      ;ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
5      ;WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
6
7 042030 177777 TIMCNT: 177777 ;WAITING COUNT
8 042032 010046 WAIT.T: MOV R0,-(SP) ;SAVE R0
9 042034 016600 000002 MOV 2(SP),R0 ;GET ADRESS OF REG. ADDESS
10 042040 010037 001204 MOV R0,$TMP3 ;WAT PC+2 IN $TMP3
11 042044 162737 000002 001204 SUB #2,$TMP3 ;WAT PC FOR TYPEOUT
12 042052 012037 001176 MOV (R0)+,$TMP0 ;WAIT REGISTER ADDRESS
13 042056 012037 001200 MOV (R0)+,$TMP1 ;WAIT ON BIT
14 042062 010066 000002 MOV R0,2(SP) ;RESTORE RETURN ON STACK
15 042066 012600 MOV (SP)+,R0 ;RESTORE R0
16 042070 013737 042030 001202 MOV TIMCNT,$TMP2 ;TEMPORARY COUNT
17 042076 033777 001200 137072 1$: BIT $TMP1,@$TMP0 ;IS REQUIRED BIT THERE?
18 042104 001021 BNE 2$ ;BRANCH IF YES
19 042106 005337 001202 DEC $TMP2 ;COUNT
20 042112 001371 BNE 1$ ;BRANCH IF NOT TIME UP
21 042114 013737 042030 001202 MOV TIMCNT,$TMP2 ;TEMPORARY COUNT
22 042122 033777 001200 137046 3$: BIT $TMP1,@$TMP0 ;IS REQUIRED BIT THERE?
23 042130 001007 BNE 2$ ;BRANCH IF YES
24 042132 005337 001202 DEC $TMP2 ;COUNT
25 042136 001371 BNE 3$ ;BRANCH IF NOT TIME UP
26 042140 017737 137032 001126 MOV @$TMP0,$BDDAT ;REGISTER CONTENTS
27 042146 104016 EMT 16
28 042150 000002 2$: RTI
29
30 ;CALL FOR THE ABOVE WAITLOOP IS
31
32 :
33 : MOV @A,X$ ;A CONTAINS REGISTER ADDRESS
34 : - - - ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
35 : - - -
36 :
37 :X$: WAT 0 ;ABSOLUTE REG. ADDRESS UNDER WAIT
38 : .WORD 0 ;BIT WAITED FOR
39 :

```

```

1      .SBTTL  SAVE ROUTINE
2
3      ;THIS IS A SUBROUTINE TO SAVE REGISTERS
4      ;IN THE REGISTER TABLE TO ANY LOCATION
5      ;THE CALL IS
6      ;JSR   R0,SAVER
7      ;FROM
8      ;TO
9      ;NUMBER OF WORDS SAVED
10
11     SAVER:
12     042152 010146      MOV     R1,-(SP)      ;;PUSH R1 ON STACK
13     042154 010246      MOV     R2,-(SP)      ;;PUSH R2 ON STACK
14     042156 010346      MOV     R3,-(SP)      ;;PUSH R3 ON STACK
15     042160 012001      MOV     (R0)+,R1      ;FROM
16     042162 012002      MOV     (R0)+,R2      ;TO
17     042164 012003      MOV     (R0)+,R3      ;NUMBER
18     042166 013122      1$:   MOV     @R1+,(R2)+  ;SAVE REGISTER CONTENTS
19     042170 005303      DEC     R3            ;COUNT
20     042172 001375      BNE    1$            ;BRANCH IF NOT DONE
21     042174 012603      MOV     (SP)+,R3      ;;POP STACK INTO R3
22     042176 012602      MOV     (SP)+,R2      ;;POP STACK INTO R2
23     042200 012601      MOV     (SP)+,R1      ;;POP STACK INTO R1
24     042202 000200      RTS     R0
25
26     .SBTTL  WRITE CHECK ROUTINE
27
28     ;THIS IS A SUBROUTINE TO DO WRITE CHECK HEADER AND DATA
29     ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
30
31     ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
32     WRCHHD: MOV     #FMT22,CYL      ;CYLINDER 0 FORMAT 16 BIT WORDS
33     MOV     #1,SECOTR+1          ;TRACK=1
34     MOV     #1,SECOTR           ;SECTOR=1
35     CLR     KEY1                 ;KEY1=0
36     CLR     KEY2                 ;KEY2=0
37     MOV     #36.,DAWORD         ;NO OF DATA WORDS
38     CLR     X                    ;THIS IS A READ OPERATION
39     JSR     R5,CRC              ;GO TO CALCULATE CRC
40     CYL
41     WCRC
42
43     ;THESE ARE REGULAR SETUPS
44     JSR     PC,CLDISK           ;SET UP GENERAL REGISTERS
45     MOV     #-40.,@RHWC         ;AND CLEAR DISK REGISTERS
46     MOV     #REINTO,@RHBA      ;36 DATA WORDS 4 HEADER WORDS
47     MOV     #1,-(SP)           ;STARTING ADDRESS OF READ BUFFER
48     MOV     #1,1(SP)           ;SECTOR=1
49     MOV     (SP)+,@RHDST       ;TRACK=1 IN UPPER BYTE
50     MOV     #FMT22!ECI,@RHOF  ;TRACK=1, SECTOR=1 IN RHDST
51     ;16 BIT WORDS
52     ;ECC CORRECTION INHIBIT BECAUSE
53     ;ECC LOGIC IS NOT CHECKED YET
    
```

53	042324	005077	136722	CLR	@RHCA	:CYLINDER=0
54	042330	004737	041514	JSR	PC,CHECKT	:CHECK THAT DVA,RDY,DPR,DRY = 1
	042334	104401	062726	TYPE	.CPHALT	:AND THAT NO OTHERS = 1. CANNOT CON-
	042340	000000		HALT		:TINUE TESTING IF BOTH AREN'T TRUE
55	042342	013711	001460	MOV	WRCHDT,@R1	:STOP THE TEST
56						:WRITE CHECK HEADER AND DATA=52
57	042346	004737	046146	JSR	PC,COMHD	:INTO RHCS1
58						:WRITE CHECK HEADER AND DATA
59						:SAME AS READ HEADER AND DATA
60	042352	000207		RTS	PC	:RETURN TO WRITE CHECK TEST


```

1      .SBTTL COMPARE ROUTINE
2
3      ;THIS IS A SUBROUTINE TO COMPARE TWO BLOCKS IN MEMORY
4      ;R1 HAS GOOD DATA BUFFER ADDRESS
5      ;R2 HAS TEST DATA BUFFER ADDRESS
6      ;$TMP0 HAS ADDRESS OF RETURN ON ERROR TO PRINT HEADER
7      ;$TMP1 HAS ADDRESS OF RETURN ON ERROR TO PRINT DATA
8      ;R3 HAS NUMBER OF WORDS TO BE COMPARED
9      ;R4 HAS ONE MORE THAN NUMBER OF WORDS TO BE COMPARED
10
11     COMPAR:
12     042354      010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
13     042356      010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
14     042360      010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
15     042362      010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
16     042364      010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
17     042366      012001      MOV      (R0)+,R1      ;ADDRESS OF GOOD DATA BUFFER
18     042370      012002      MOV      (R0)+,R2      ;ADDRESS OF TEST DATA BUFFER
19     042372      012003      MOV      (R0)+,R3      ;NO OF WORDS TO BE COMPARED
20     042374      012037      001176    MOV      (R0)+,$TMP0    ;RETURN ON ERROR TO PRINT HEADER
21     042400      012037      001200    MOV      (R0)+,$TMP1    ;RETURN ON ERROR TO PRINT DATA
22     042404      011000      MOV      (R0),R0       ;RETURN ON NO ERROR
23     042406      010304      MOV      R3,R4        ;NO OF WORDS TO BE COMPARED
24     042410      005204      INC      R4
25     042412      010437      046426    1$:     MOV      R4,ERWORD     ;FOR ERROR WORD NO
26     042416      022122      CMP      (R1)+,(R2)+   ;COMPARE GOOD WITH TEST DATA
27     042420      001426      BEQ      3$           ;BRANCH IF GOOD
28
29     042422      014137      001124    MOV      -(R1),$GDDAT  ;GOOD DATA
30     042426      014237      001126    MOV      -(R2),$BDDAT  ;BAD DATA
31     042432      160337      046426    SUB      R3,ERWORD     ;ERROR WORD NO.
32     042436      005737      001406    TST      ERFLG$ ;ANY ERRORS ALREAY THERE
33     042442      001003      BNE      2$           ;BRANCH IF YES
34     042444      004777      136526    JSR      PC,@$TMP0     ;RETURN TO PRINT HEADER
35     042450      000402      BR       5$           ;BRANCH TO AVOID PRINTING NEXT ERROR
36     042452      004777      136522    2$:     JSR      PC,@$TMP1     ;RETURN TO PRINT DATA
37     042456      022122      5$:     CMP      (R1)+,(R2)+   ;UNDO -(R1) AND -(R2) FOR ERRORS
38     042460      017746      136454    MOV      @SWR,-(SP)    ;GET SWITCH SETTING
39     042464      042716      177177    BIC      #^C600,(SP)   ;KEEP ONLY SWITCH 7 AND 8
40     042470      022726      000200    CMP      #SW07,(SP)+   ;IS 7 SET AND 8 RESET
41     042474      001402      4$:     BEQ      4$           ;BRANCH OUT IF YES
42     042476      005303      3$:     DEC      R3           ;COUNT
43     042500      001344      1$:     BNE      1$           ;BRANCH IF ALL NOT DEVICE
44
45     042502      012605      4$:     MOV      (SP)+,R5      ;;POP STACK INTO R5
46     042504      012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
47     042506      012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
48     042510      012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
49     042512      012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
50     042514      000200      RTS      R0           ;RETURN TO MAIN PROGRAM
51
52     ;THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
53     ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
54
55
56
57

```

```

48                                     ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
49
50 042516 012737 010000 046306 WRCHDA: MOV    #FMT22,CYL      ;CYLINDER 0 FORMAT 16 BIT WORDS
51 042524 112737 000001 046311      MOVB   #1,SECOTR+1    ;TRACK=1
52 042532 112737 000001 046310      MOVB   #1,SECJTR     ;SECTOR=1
53 042540 005037 046312                CLR    KEY1          ;KEY1=0
54 042544 005037 046314                CLR    KEY2          ;KEY2=0
55 042550 012737 000040 046366      MOV    #32.,DAWORD   ;NO OF DATA WORDS
56 042556 005037 046316                CLR    X             ;THIS IS A READ OPERATION
57
58 042562 004537 042666                JSR    R5,CRC        ;GO TO CALCULATE CRC
59 042566 046306
60 042570 050206                WCRD
61
62                                     ;THESE ARE REGULAR SETUPS
63
64 042572 004737 041460                JSR    PC,CLDISK    ;SET UP GENERAL REGISTERS
65                                     ;AND CLEAR DISK REGISTERS
66
67 042576 012777 177740 136426      MOV    #-32.,@RHWC   ;36 DATA WORDS 4 HEADER WORDS
68 042604 012777 002554 136422      MOV    #REINT0,@RHBA ;STARTING ADDRESS OF READ BUFFER
69 042612 112746 000001                MOVB   #1,-(SP)      ;SECTOR=1
70 042616 112766 000001 000001      MOVB   #1,1(SP)     ;TRACK=1 IN UPPER BYTE
71 042624 012677 136414                MOV    (SP)+,@RHDST ;TRACK=1, SECTOR=1 IN RHDST
72 042630 012777 014000 136412      MOV    #FMT22!ECI,@RHOF ;16 BIT WORDS
73                                     ;ECC CORRECTION INHIBIT BECAUSE
74                                     ;ECC LOGIC IS NOT CHECKED YET
75 042636 005077 136410                CLR    @RHCA        ;CYLINDER=0
76 042642 004737 041514                JSR    PC,CHECKT    ;CHECK THAT DVA,RDY,DPR,DRY = 1
    042646 104401 062726                TYPE   ,CPHALT     ;AND THAT NO OTHERS = 1. CANNOT CON-
    ;TINUE TESTING IF BOTH AREN'T TRUE
    ;STOP THE TEST
    042652 000000                HALT
77 042654 013711 001456                MOV    WRCHK,@R1    ;WRITE CHECK DATA=50 INTO RHCS1
78 042660 004737 046146                JSR    PC,COMHD     ;WRITE CHECK HEADER AND DATA
79                                     ;SAME AS READ HEADER AND DATA
80
81 042664 000207                RTS    PC           ;RETURN TO WRITE CHECK TEST
  
```

```

1      .SBTTL  CRC GENERATION ROUTINE
2
3      ;THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
4      ;HEADER WORDS AND STORE THEM IN 'WCRC' AND 'GCRC'
5      ;R1 - REGISTER FOR CRC, INCREMENTED CRC VALUE IS HERE
6      ;R2 - THIS HAS BIT POSITION 2 VALUE C
7      ;R3 - THIS HAS BIT POSITION 16 I.E. OUTPUT BIT VALUE B
8      ;R4 - THIS HAS BIT POSITION 15 VALUE E
9      ;$TMP0 - NUMBER OF WORDS
10     ;$TMP2 - NUMBER OF BITS PER WORD = 16
11     ;$TMP3 - TEMPORARY REG.
12     ;$TMP4 - TEMPORARY REG TO TRANSFER CARRY
13     ;$TMP5 - THIS HAS DATA BIT VALUE D
14
15     ;FETCH DATA BIT D
16     ;B = D XOR 16
17     ;C = B XOR 2
18     ;E = B XOR 15
19     ;ROTATE RIGHT ONE POSITION
20     ;B GOES TO POSITION 1
21     ;C GOES TO POSITION 3
22     ;E GOES TO POSITION 16
23     ;REPET 64 TIMES
24     ;CALL   JSR     R5,CRC
25     ;X      ;FIRST LOCATION AT
26     ;Y      ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
27
28     CRC:
29     042666 010046      MOV     R0,-(SP)      ;;PUSH R0 ON STACK
30     042670 012500      MOV     (R5)+,R0     ;GET POINTER TO CYL NO.
31     042672 010146      MOV     R1,-(SP)      ;;PUSH R1 ON STACK
32     042674 010246      MOV     R2,-(SP)      ;;PUSH R2 ON STACK
33     042676 010346      MOV     R3,-(SP)      ;;PUSH R3 ON STACK
34     042700 010446      MOV     R4,-(SP)      ;;PUSH R4 ON STACK
35     042702 005001      CLR     R1            ;CLEAR WORKING LOCATION
36     042704 005037 001210  CLR     $TMP5
37     042710 012737 000004 001176  MOV     #4,$TMP0      ;WORD COUNT
38     042716 012037 001204 16$:  MOV     (R0)+,$TMP3   ;TEMPORARY WORD STORAGE
39     042722 012737 000020 001202  MOV     #16,$TMP2    ;BIT COUNT
40     042730 013737 001204 001206  MOV     $TMP3,$TMP4  ;TEMPORARY WORD STORAGE
41     042736 006037 001204 15$:  ROR     $TMP3         ;GET LSB INTO 'C'
42     042742 006037 001210      ROR     $TMP5         ;GET ABOVE 'C' INTO $TMP5
43     042746 032701 000001      BIT     #BIT0,R1      ;IS POSITION 15 HIGH
44     042752 001403      BEQ     1$           ;BRANCH IF POSITION 16 LOW
45     042754 012703 100000      MOV     #BIT15,R3    ;GET POSITION 16
46     042760 000401      BR     2$           ;GET POSITION 16
47     042762 005003 1$:  CLR     R3          ;XOR POSITION 16 WITH D
48     042764 063703 2$:  ADD     $TMP5,R3    ;TO GIVE B
49     042770 032701 040000      BIT     #BIT14,R1    ;IS POSITION 2 HIGH
50     042774 001403      BEQ     3$           ;BRANCH IF POSITION 2 LOW
51     042776 012702 100000      MOV     #BIT15,R2    ;GET POSITION 2
52     043002 000401      BR     4$           ;GET POSITION 2
53     043004 005002 3$:  CLR     R2          ;XOR B WITH POSITION 2
54     043006 060302 4$:  ADD     R3,R2       ;TO GIVE C
55     043010 032701 000002      BIT     #BIT1,R1     ;IS POSITION 15 HIGH

```

54	043014	001403		BEO	5\$:BRANCH IF POSITION 15 LOW
55	043016	012704	100000	MOV	#BIT15,R4	:GET POSITION 15
56	043022	000401		BR	6\$	
57	043024	005004		5\$: CLR	R4	:GET POSITION 15
58	043026	060304		6\$: ADD	R3,R4	:XOR POSITION 15 WITH B
59						:TO GIVE E
60	043030	006037	001206	ROR	\$TMP4	:GET LSB INTO 'C'
61	043034	006001		ROR	R1	:GET ABOVE C INTO R1
62	043036	005703		TST	R3	:TEST B
63	043040	100403		BMI	7\$:BRANCH IF B=1
64	043042	042701	100000	BIC	#BIT15,R1	:SET B IN POSITION 1
65	043046	000402		BR	10\$	
66	043050	052701	100000	7\$: BIS	#BIT15,R1	:SET B IN POSITION 1
67	043054	005702		10\$: TST	R2	:TEST C
68	043056	100403		BMI	11\$:BRANCH IF C=1
69	043060	042701	020000	BIC	#BIT13,R1	:GET C IN POSITION 3
70	043064	000402		BR	12\$	
71	043066	052701	020000	11\$: BIS	#BIT13,R1	:GET C IN POSITION 3
72	043072	005704		12\$: TST	R4	:TEST E
73	043074	100403		BMI	13\$:BRANCH IF E=1
74	043076	042701	000001	BIC	#BIT0,R1	:GET E IN POSITION 16
75	043102	000402		BR	14\$	
76	043104	052701	000001	13\$: BIS	#BIT0,R1	:GET E IN POSITION 16
77	043110	005337	001202	14\$: DEC	\$TMP2	:BIT COUNTER
78	043114	001310		BNE	15\$:BRANCH IF 16 NOT DONE
79	043116	005337	001176	DEC	\$TMP0	:WORD COUNTER
80	043122	001275		BNE	16\$:BRANCH IF 4 NOT DONE
81	043124	010135		MOV	R1,a(R5)+	:PUT CRC WHERE DESIRED
82	043126	012604		MOV	(SP)+,R4	::POP STACK INTO R4
	043130	012603		MOV	(SP)+,R3	::POP STACK INTO R3
	043132	012602		MOV	(SP)+,R2	::POP STACK INTO R2
	043134	012601		MOV	(SP)+,R1	::POP STACK INTO R1
	043136	012600		MOV	(SP)+,R0	::POP STACK INTO R0
83	043140	000205		RTS	R5	
84						
85						:THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
86						:CYLINDER 0 (16 BITS PER WORD)
87						:TRACK 1, SECTOR 1
88						:KEY1 1
89						:KEY2 1
90						:CRC THROUGH THE JSR R5,CRC
91						:256 WORDS OF 177400
92						
93						:CALL JSR PC,SETDSK
94						
95	043142			SETDSK:		
	043142	010046		MOV	R0,-(SP)	::PUSH R0 ON STACK
	043144	010146		MOV	R1,-(SP)	::PUSH R1 ON STACK
	043146	010246		MOV	R2,-(SP)	::PUSH R2 ON STACK
96	043150	012700	177400	MOV	#177400,R0	:DATA IN THE DISK
97	043154	012701	000400	MOV	#256.,R1	:COUNTER
98	043160	012702	050224	MOV	#DISK,R2	:START OF SIMULATOR DISK
99	043164	010022		1\$: MOV	R0,(R2)+	:MOVE IN DATA
100	043166	005301		DEC	R1	:COUNT FOR 256
101	043170	001375		BNE	1\$:BRANCH IF 256 NOT COMPLETE
102	043172	012701	000021	MOV	#17.,R1	:2 ECC WORDS, 1 DATA GAP
103						:14 TOLERANCE GAP

```
104 043176 005022      2$: CLR (R2)+      ;CLEAR ECC,DATA GAP AND
105                    ;TOLERANCE GAP
106 043200 005301      DEC R1             ;COUNT
107 043202 001375      BNE 2$           ;BRANCH IF NOT COMPLETE
108
109                    ;NOW SET UP FOR DISKLESS USE
110
111 043204 012737 010000 046306      MOV #FMT22,CYL    ;CYLINDER 0 (16 BIT WORDS)
112 043212 112737 000001 046311      MOV# #1,SECOTR+1 ;TRACK=1
113 043220 112737 000001 046310      MOV# #1,SECOTR   ;SECTOR=1
114 043226 012737 000001 046312      MOV #1,KEY1 ;KEY1=1
115 043234 012737 000001 046314      MOV #1,KEY2 ;KEY2=1
116 043242 013737 000400 046366      MOV 256,DAWORD  ;NO. OF DATA WORDS
117 043250 004537 042666      JSR R5,CRC ;GO TO CALCULATE CRC
118 043254 046306      CYL              ;FIRST CRC WORD
119 043256 050206      WCRC           ;PUT CALCULATED CRC
120 043260 012602      MOV (SP)+,R2    ;:POP STACK INTO R2
      043262 012601      MOV (SP)+,R1    ;:POP STACK INTO R1
      043264 012600      MOV (SP)+,R0    ;:POP STACK INTO R0
121 043266 000207      RTS PC
```

```

1
2
3      ;THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
4      ;(BIT #7) AND CRC ERROR (BIT #8)
5
6      ;CALL JSR RO,HCCRCE
7
8      :      COM      ;COMMAND-READ HEADER AND DATA
9      :      :      -WRITE DATA
10     :      C      ;CYLINDER
11     :      S      ;SECTOR
12     :      T      ;TRACK
13     :      -N.    ;WORD COUNT
14     :      B      ;RHBA BUFFER START
15     :      X      ;1=WRITE DATA 0=READ
16     :      H      ;H=1 HEADER CHECK, H=0 CRC CHECK
17 043270 010037 001414 HCCRCE: MOV RO,PCJSR ;SAVE PC OF JSR+4
18 043274 162737 000004 001414 SUB #4,PCJSR ;GET PC OF JSR
19 043302 004737 041460 JSR PC,CLDISK ;INIT AND SETUP GENERAL REG.
20 043306 004737 041514 JSR PC,CHECKT ;CHECK THAT DVA,RDY,DPR,DRY = 1
    043312 104401 062726 TYPE ,CPHALT ;AND THAT NO OTHERS = 1. CANNOT CON-
    ;TINUE TESTING IF BOTH AREN'T TRUE
    ;STOP THE TEST
21 043316 000000 HALT ;SAVE COMMAND
22 043320 011037 001210 MOV (RO),$TMP5 ;COMMAND
23 043324 012011 MOV (RO)+,@R1 ;CYLINDER
24 043326 012077 135720 MOV (RO)+,@RHCA ;SECTOR
25 043332 112046 MOV (RO)+,-(SP) ;UP DATE RO
26 043334 105720 TSTB (RO)+ ;TRACK
27 043336 112066 000001 MOV (RO)+,1(SP) ;UPDATE RO
28 043342 *05720 TSTB (RO)+ ;TRACK SECTOR
29 043344 012677 135674 MOV (SP)+,@RHST ;NO. OF DATA WORDS +4 HEADER
30 043350 012077 135656 MOV (RO)+,@RHWC ;IF A READ HEADER AND DATA
31 043354 012077 135654 MOV (RO)+,@RHBA ;STARTING ADDRESS OF BUFFER
32 043360 012037 046316 MOV (RO)+,X ;X=0 READ HEADER AND DATA
33 ;X=1 WRITE DATA
34 043364 012777 014000 135656 MOV #FMT22!ECI,@RHOF ;16 BITS PER WORD
35 ;ECC CORRECTION INHIBIT
36 043372 005037 001406 CLR ERFLG$ ;CLEAR ERROR FLAG
37 043376 004737 046146 JSR PC,COMHD ;COMMAND
38
39 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
40 ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
41 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
42 ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
43 ;DETECTED
44 ;HEADER AND DATA ARE TO BE CHECKED.
45
46 043402 004737 041064 JSR PC,PUTREG ;SAVE REGISTERS
47 043406 005737 001406 TST ERFLG$ ;ANY ERRORS ALREADY THERE
48 043412 001034 BNE 10$ ;BRANCH IF YES
49 043414 005737 046316 TST X ;IS THIS A READ
50 043420 001015 BNE 3$ ;IF A WRITE DATA BRANCH

```

```

1
2
3
4
5
6
7
8 043422 004037 042354 JSR RO,COMPAR ;CHECK
9 043426 001510 WRFROM ;GOOD DATA
10 043430 002554 REINTO ;TEST BUFFER
11 043432 000400 256. ;4 HEADER 252 DATA
12 043434 043442 1$ ;RETURN POINT FOR ERROR HEADER
13 043436 043446 2$ ;RETURN POINT FOR ERROR DATA
14 043440 043504 10$ ;RETURN FOR GOOD COMPARISON
15 043442 1$:
16 043444 104004 EMT 4
17 043446 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
18 043446 104005 2$: EMT 5
19
20
21
22
23
24 043450 000207 RTS PC ;HEADER WORDS AND HENCE
25
26 043452 000414 BR 10$ ;SHOULD BE READ AS WRITTEN ON
27 ;DISK, WORD NOS. 5 ONWARDS
28 ;SHOULD NOT BE READ AND HENCE
29 ;READ INTO BUFFER
30 ;SHOULD BE UNCHANGED
31 ;RETURN TO COMPARISON
32
33
34 043454 004037 042354 3$: JSR RO,COMPAR ;CHECK
35 043460 002554 REINTO ;GOOD DATA BUFFER
36 043462 050224 DISK ;TEST BUFFER
37 043464 000400 256.
38 043466 043474 4$ ;RETURN POINT FOR ERROR HEADER
39 043470 043500 5$ ;RETURN POINT FOR ERROR DATA
40 043472 043504 10$ ;RETURN POINT FOR GOOD COMPARISON
41 043474 4$:
42 043476 104004 EMT 4
43 043500 000207 RTS PC ;RETURN TO COMPARISON SUBROUTINE
44 043500 104005 5$: EMT 5
45
46
47 043502 000207
48 043504 005720 10$: RTS PC ;WORDS THE SHOULD NOT
49 043506 001442 TST (R0)+ ;HAVE BEEN CHANGED BY THE
50 043510 022737 000072 001210 BEQ 6$ ;WRITE COMMAND
51 043516 001417 CMP #72,$TMP5 ;RETURN TO COMPARISON SUBROUTINE
52 043520 017737 135516 001126 BEQ 11$ ;IS THIS A HCRC ON HCE CHECK?
53 043526 022737 000200 001126 MOV @RHER1,$BDDAT ;BRANCH IF HCRC
;IS THIS A READ COMMAND
;BRANCH IF YES
;TEST DATA
;ONLY HEADER COMPARE BIT?

```



```

109 043754 052777 000010 135254      BIS      #BAI,@RHCS2      ;BUS ADDRESS INCREMENT INHIBIT
110 043762 012777 010000 135260      MOV      #FMT22,@RHOF  ;FORMAT 16 BIT WORDS
111 043770 005077 135256      CLR      @RHCA         ;CYLINDER=0
112 043774 012737 000001 044022      MOV      #1,MID ;SECTOR IS SET TO 1 SO THAT
113                                     ;WE CAN GET OUT AT THE
114                                     ;MIDDLE OF AN OPERATION
115                                     ;LOOKING FOR SECTOR 10
116 044002 012777 000001 135250      MOV      #DMD,@RHMR    ;SET DIAGNOSTIC MODE
117 044010 052777 000001 135222      BIS      #GO,@RHCS1    ;GO TO RHCS1 WITH 62
118 044016 004137 052354      JSR      R1,SEARCH
119 044022 000000      MID:    .WORD      0      ;SECTOR
120 044024 012601      MOV      (SP)+,R1      ;:POP STACK INTO R1
121 044026 012600      MOV      (SP)+,R0      ;:POP STACK INTO R0
122 044030 000207      RTS      PC
123
124      .SBTTL  JAM CURRENT CYLINDER ROUTINE
125      ;THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
126      ;THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
127      ;WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE
128      ;
129      ;CALL
130      ;
131      JSR      R0,MAKECYL
132      .WORD      XC      ;DESIRED VALUE OF CURRENT CYLINDER
133
134      MAKECYL:
135      MOV      R5,-(SP)  ;:PUSH R5 ON STACK
136      MOV      R0,PCJSR ;:PC OF JSR+4
137      SUB      #4,PCJSR ;:SAVE PC OF JSR
138      MOV      (R0)+,R5 ;:GETTING READY TO FILL DESIRED CYLINDER
139      MOV      R5,@RHCA ;:FILL DESIRED CYLINDER REGISTER
140      CLR      @RHDST   ;:MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
141      MOV      SEECOM,@RHCS1 ;:FILL SEEK COMMAND
142      MOV      #DMD,@RHMR ;:SET DIAGNOSTIC MODE
143      BIS      #GO,@RHCS1 ;:GO TO SEEK
144      NOP
145      NOP              ;:ALLOW TIME FOR SEEK TO HANG UP
146      NOP              ;:ALLOW TIME FOR SEEK TO HANG UP
147      NOP              ;:ALLOW TIME FOR SEEK TO HANG UP
148      JSR      PC,CLDISK ;:GIVE INIT
149      MOV      @RHCC,$BDDAT ;:TEST DATA
150      CMP      R5,$BDDAT ;:COMPARE CURRENT CYLINDER
151      BEQ      1$      ;:BRANCH IF GOOD
152      MOV      R5,$GDDAT ;:GOOD VALUE OF RHCC
153      MOV      RHCC,REGADR ;:FAILING REGISTER ADDRESS
154      EMT      30
155      ;REGISTER AFTER A SEEK AND AN INIT
156      1$:
157      MOV      (SP)+,R5  ;:POP STACK INTO R5
158      RTS      R0
159
160      .SBTTL  ECC GENERATION AND COMPARISON ROUTINE
161      ;THIS SUBROUTINE GENERATES AND TESTS FCC
162      ;CALL
163      ;
164      JSR      PC,ECTEST
    
```

164	100000	PIE1	=100000
165	040000	PIE2	=40000
166	020000	PIE3	=20000
167	010000	PIE4	=10000
168	004000	PIE5	=4000
169	002000	PIE6	=2000
170	001000	PIE7	=1000
171	000400	PIE8	=400
172	000200	PIE9	=200
173	000100	PIE10	=100
174	000040	PIE11	=40
175	000020	PIE12	=20
176	000010	PIE13	=10
177	000004	PIE14	=4
178	000002	PIE15	=2
179	000001	PIE16	=1
180	100000	PIE17	=100000
181	040000	PIE18	=40000
182	020000	PIE19	=20000
183	010000	PIE20	=10000
184	004000	PIE21	=4000
185	002000	PIE22	=2000
186	001000	PIE23	=1000
187	000400	PIE24	=400
188	000200	PIE25	=200
189	000100	PIE26	=100
190	000040	PIE27	=40
191	000020	PIE28	=20
192	000010	PIE29	=10
193	000004	PIE30	=4
194	000002	PIE31	=2
195	000001	PIE32	=1

196				
197	044152	000000	ECDATA: 0	:DATA BIT FOR ECC :IF ALL ONES THEN CURRENT BIT IS A ONE :IF ZERO THEN CURRENT BIT IS A ZERO
198				
199				
200				
201	044154	000000	GECC1: 0	:LOW ORDER ECC WORD TO BE GENERATED HERE :=R1
202				
203				
204	044156	000000	GECC2: 0	:HIGH ORDER ECC WORD TO BE GENERATED HERE :=R2
205				
206				
207	044160	000000	TSECCG: 0	:IF =177777 GENERATE AND TEST ECC FOR THIS BIT :IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT
208				
209				
210	044162	113713	NCODE: 38859.	:N-CODE WORD
211	044164	000000	NCOUNT: 0	:TEMPORARY N CODE
212	044166	000000	POSITI: 0	:POSITION REGISTER
213	044170	010041	HARDER: 4129.	:HARD ERROR COUNT :TRUE COUNT IS 4128 BUT AS COMPARES ARE :DONE ONE STAGE LATER SO 4129
214				
215				
216	044172	000000	DATENV: 0	:DATA ENVELOPE FOR TYPE OUT :MAX FOR WRITE IS 4096 :MAX FOR READ IS 4128
217				
218				
219	044174	000000	ZCODE: 0	:LEADING ZEROS ENVELOPE FOR TYPE OUT :THIS IS SHUT OFF WHEN POSITION COUNTER
220				

```

221                                     ;IN ENABLED
222                                     ;MAX COUNT IS 38859
223
224
225
226 044176 000000      HADTMP: 0          ;TEMPORARY HARD ERROR COUNT
227 044200 000000      P3: 0
228 044202 000000      P12: 0
229 044204 000000      P22: 0
230 044206 000000      P24: 0
231
232
233
234
235
236 044210      ECTEST:
    044210 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
    044212 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
    044214 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
    044216 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
    044220 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
    044222 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
237 044224 013701 044154      MOV      GECC1,R1      ;ECC1 WORD
238 044230 013702 044156      MOV      GECC2,R2      ;ECC2 WORD
239 044234 005737 044152      TST      ECDATA      ;IS CURRENT BIT A ONE
240 044240 001406      BEQ      2$          ;BRANCH IF CURRENT DATA D=0
241
242                                     ;IF CARRY IS NOT ZERO THEN D=1
243                                     ;INVERT X32 TO GIVE R0
244
245 044242 010103      1$:  MOV      R1,R3
246 044244 052703 177776      BIS      #^CPIE32,R3
247 044250 005103      COM      R3
248 044252 010300      MOV      R3,R0
249 044254 000404      BR      3$
250
251                                     ;IF CARRY IS ZERO THEN D=0
252                                     ;X32 BECOMES R0
253 044256 010103      2$:  MOV      R1,R3
254 044260 042703 177776      BIC      #^CPIE32,R3
255 044264 010300      MOV      R3,R0
256
257 044266 000241      3$:  CLC
258 044270 006000      ROR      R0
259 044272 006000      ROR      R0
260 044274 005700      TST      R0
261 044276 001462      BEQ      10$          ;BRANCH IF R0=0
262
263                                     ;INVERT X2
264 044300 010203      MOV      R2,R3
265 044302 052703 137777      BIS      #^CPIE2,R3
266 044306 005103      COM      R3
267 044310 010337 044200      MOV      R3,P3
268 044314 006237 044200      ASR      P3
269
270                                     ;INVERT X11
271

```

```

272
273 044320 010203          MOV      R2,R3
274 044322 052703 177737  BIS      #^CPIE11,R3
275 044326 005103          COM      R3
276 044330 010337 044202  MOV      R3,P12
277 044334 006237 044202  ASR      P12
278
279          :INVERT X21
280
281 044340 010103          MOV      R1,R3
282 044342 052703 173777  BIS      #^CPIE21,R3
283 044346 005103          COM      R3
284 044350 010337 044204  MOV      R3,P22
285 044354 006237 044204  ASR      P22
286
287          :INVERT X23
288
289 044360 010103          MOV      R1,R3
290 044362 052703 176777  BIS      #^CPIE23,R3
291 044366 005103          COM      R3
292 044370 010337 044206  MOV      R3,P24
293 044374 006237 044206  ASR      P24
294
295          :NOW THAT R0 FOR POSITION 1
296          :          P3 FOR POSITION 3
297          :          P12 FOR POSITION 12
298          :          P22 FOR POSITION 22
299          :          P24 FOR POSITION 24
300          :ARE KNOWN THE ROTATE WILL BE DONE AND
301          :THESE BITS JAMED IN
302
303 044400 006002          ROR      R2
304 044402 006001          ROR      R1
305 044404 053700 044200  BIS      P3,R0
306 044410 053700 044202  BIS      P12,R0
307 044414 042702 120020  BIC      #PIE1!PIE3!PIE12,R2
308 044420 050002          BIS      R0,R2
309
310 044422 005000          CLR      R0
311 044424 053700 044204  BIS      P22,R0
312 044430 053700 04.206  BIS      P24,R0
313 044434 042701 002400  BIC      #PIE22!PIE24,R1
314 044440 050001          BIS      R0,R1
315 044442 000404          BR      12$
316
317          :THE PROGRAM COMES HERE IF R0=0
318          :SO AFTER ROTATE R0 GETS PUT INTO POSITION 1
319
320 044444 006002          10$: ROR      R2
321 044446 006001          ROR      R1
322 044450 042702 100000  BIC      #PIE1,R2
323 044454 010137 044154  12$: MOV      R1,GECC1          :SAVE ECC1
324 044460 010237 044156  MOV      R2,GECC2          :SAVE ECC2
325 044464 005737 044160  TST      TSECCG          :IS HARDWARE TO BE CHECKED
326          :IF =1777777 TEST HARDWARE
327          :IF = 0 DO NOT TEST HARDWARE
328 044470 001432          BEQ      14$          :BRANCH IF HARDWARE NOT TO BE CHECKED
    
```

```

329
330 ;CHECK HARDWARE
331 044472 032777 000400 134440 BIT #SW8,@SWR ;IS SWITCH 8 SET
332 044500 001005 BNE 15$ ;BRANCH IF SW8 IS SET
333 044502 032777 000100 134430 BIT #SW6,@SWR ;IS SWITCH 6 SET
334 044510 001401 BEQ 15$ ;BRANCH IF SW6 IS NOT SET
335 044512 000421 BR 14$ ;IF SWITCH 8 IS NOT SET AND
336 ;SWITCH 6 IS SET THEN
337 ;DO NOT DO COMPARES
338 044514 010146 15$: MOV R1,-(SP) ;GOOD PATTERN REGISTER
339 044516 042716 174000 BIC #174000,(SP) ;GET ONLY PATTERN BITS
340 044522 022677 134544 CMP (SP)+,@RHEC2 ;COMPARE PATTERN REGISTER
341 044526 001404 BEQ 13$ ;BRANCH IF GOOD
342 ;TO SAVE TIME
343 044530 004737 041064 JSR PC,PUTREG ;SAVE REGISTERS
344 044534 104035 EMT 35
345 044536 000407 BR 14$ ;BRANCH OUT
346 044540 023777 044166 13$: CMP POSITI,@RHEC1 ;COMPARE POSITION REGISTER
347 044546 001403 BEQ 14$ ;BRANCH IF GOOD
348 ;TO SAVE TIME
349 044550 004737 041064 JSR PC,PUTREG ;SAVE REGISTERS
350 044554 104035 EMT 35
351 ;'DATA ENVLOP' GIVES NUMBER OF CLOCK
352 ;PULSES FROM BEGINING OF COMMAND
353 ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
354 ;
355 ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
356 ;IN A READ THERE ARE 10040 OCTAL CLOCKS
357 ;
358 ;
359 ;'N-CODE ZEROS' GIVE THE NUMBER OF CLOCKS
360 ;GIVEN FOR THE LEADING ZEROS FIELD
361 ;MAX COUNT IS 113713 OCTAL
362 ;
363 ;'GOOD POSITION' GIVES NUMBER OF CLOCKS
364 ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
365 ;FIELD
366 ;MAX COUNT IS 10040 OR 10041 OCTAL
367
368
369 044556 012605 14$: MOV (SP)+,R5 ;;POP STACK INTO R5
044556 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
044562 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
044564 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
044566 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
044570 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
370 044572 000207 RTS PC
    
```

```

1
2
3      ;THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
4      ;FOR ERROR CORRECTION PROCESS
5      ;CALL
6      ;
7      ;
8      ;
9      044574 000000      ERPOS: 0      ;POSITION REG. WHEN CORRECTION IS COMPLETE
10
11 044576 010037 001414      ECORR: MOV R0,PCJSR      ;SAVE PC OF JSR + 4
12 044602 162737 000004 001414      SUB #4,PCJSR      ;SAVE PC OF JSR
13 044610 012037 044574      MOV (R0)+,ERPOS      ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
14 044614 010146      MOV R1,-(SF)      ;PUSH R1 ON STACK
15 044616 013701 001260      MOV RHMR,R1 ;MAINTENANCE REGISTER
16 044622 012711 000001      MOV #DMD,@R1      ;SET DIAGNOSTIC MCDE BIT
17 044626 005037 044152      CLR ECDATA ;ECC DATA IS ZERO
18
19
20
21 044632 005737 044166      1$: TST POSITI ;IS SOFTWARE POSITION NON ZERO
22 044636 001007      BNE 2$ ;BRANCH IF N-CODE S COMPLETE
23 044640 005337 044164      DEC NCOUNT ;DECREMENT N-CODE
24 044644 001001      BNE 6$ ;BRANCH IF N-CODE IS NOT COMPLETE
25 044646 000403      BR 2$ ;BRANCH AS N-CODE IS COMPLETE
26 044650 005237 044174      6$: INC ZCODE ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
27 044654 000420      BR 3$ ;BRANCH AS N-CODE IS NOT COMPLETE
28 ;GO TO GIVE CLOCK AND TEST ECC
29 044656 005237 044166      2$: INC POSITI ;INCREMENT SOFTWARE POSITION
30 044662 023737 044574 044166      CMP ERPOS,POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
31 044670 103012      BHS 3$ ;BRANCH IF MORE CLOCKS TO BE GIVEN
32 044672 023737 044176 044166      CMP HADTMP,POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
33 ;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
34 044700 001415      BEQ 5$ ;BRANCH IF YES
35 044702 032711 000400      BIT #ZER,@R1 ;CHECK ZERO DETECT BIT IN RHMR
36 044706 001016      BNE 4$ ;BRANCH IS ZER SET
37 ;TO SAVE TIME
38 044710 004737 041064      JSR PC,PUTREG ;SAVE REGISTERS
39 044714 104034      EMT 34
40 ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0
41
42
43 044716 052711 000002      3$: BIS #MCLK,@R1 ;SET CLOCK
44 044722 042711 000002      BIC #MCLK,@R1 ;CLEAR CLOCK
45 044726 004737 044210      JSR PC,ECTEST ;GO TO GENERATE AND TEST ECC
46 044732 000737      BR 1$ ;CONTINUE
47
48 ;THIS EXTRA CLOCK IS TO BRING ECH HIGH
49 ;AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL
50
51 044734 052711 000002      5$: BIS #MCLK,@R1 ;SET CLOCK
52 044740 042711 000002      BIC #MCLK,@R1 ;CLEAR CLOCK
53
54 044744      4$: MOV (SP)+,R1 ;:POP STACK INTO R1
55 044746 000200      RTS R0
56
    
```

```

57      ;THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
58      ;ON LOCATIONS 'DISK+1000' AND 'DISK+1002'
59
60 044750      FILLEC:
044750      010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
044752      010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
044754      010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
044756      010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
044760      010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
044762      010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
61 044764      005037      044166      CLR      POSITI ;CLEAR POSITION
62 044770      005037      044154      CLR      GECC1  ;CLEAR GECC1
63 044774      005037      044156      CLR      GECC2  ;CLEAR
64 045000      012701      050224      MOV      #DISK,R1 ;POINTER TO DATA FOR ECC GENERATION
65 045004      012702      000400      MOV      #256.,R2 ;COUNTER FOR NUMBER OF DATA WORDS
66 045010      012703      000020      9$: MOV      #16.,R3 ;COUNTER FOR NUMBER OF BITS PER WORD
67 045014      012104      MOV      (R1)+,R4 ;DATA IN R4
68 045016      006004      10$: ROR      R4 ;GET ONE DATA BIT IN CARRY
69 045020      103004      BCC      11$ ;BRANCH IF DATA BIT IS ZERO
70 045022      012737      177777      044152      MOV      #-1,ECDATA ;ECC DATA BIT IS A ONE
71 045030      000402      BR      12$ ;BRANCH TO GENERATE ECC
72 045032      005037      044152      11$: CLR      ECDATA ;ECC DATA BIT IS A ZERO
73 045036      004737      044210      12$: JSR      PC,ECTEST ;GO TO GENERATE ECC
74 045042      005303      DEC      R3 ;DECREMENT BIT COUNT
75 045044      001364      BNE      10$ ;BRANCH IF 16 BITS NOT DONE
76 045046      005302      DEC      R2 ;DECREMENT WORD COUNT
77 045050      001357      BNE      9$ ;BRANCH IF 256 WORDS NOT DONE
78 045052      013737      044154      051224      MOV      GECC1,DISK+<256.*2>;INSERT ECC1 ON DISK
79 045060      013737      044156      051226      MOV      GECC2,DISK+<257.*2>;INSERT ECC2 ON DISK
80 045066      012605      MOV      (SP)+,R5 ;;POP STACK INTO R5
045070      012604      MOV      (SP)+,R4 ;;POP STACK INTO R4
045072      012603      MOV      (SP)+,R3 ;;POP STACK INTO R3
045074      012602      MOV      (SP)+,R2 ;;POP STACK INTO R2
045076      012601      MOV      (SP)+,R1 ;;POP STACK INTO R1
045100      012600      MOV      (SP)+,R0 ;;POP STACK INTO R0
81 045102      000207      RTS      PC
  
```

```

1      .SBTTL  RH BASE ADDRESS CHANGE ROUTINE
2
3      :THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
4      :ADDRESS FROM 176700 TO ANY TYPED VALUE
5
6 045104 012706 001100  BASECH: MOV  #STACK,SP      ;RESET STACK
7 045110 104401 045116  TYPE  ,65$          ;::TYPE ASCIZ STRING
   045114 000415  BR      64$          ;::GET OVER THE ASCIZ
   045150  ;:65$: .ASCIZ <CRLF>/PRESENT BASE ADDRESS IS /
8 045150 013746 001240  ;:64$:
9 045154 104402  MOV  RHCS1,-(SP)      ;GET READY TO TYPE OLD BASE
10 045156 104401 045164  TYPOC
   045162 000425  TYPE  ,67$          ;::TYPE ASCIZ STRING
   ;:67$: .ASCIZ <CRLF>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /
11 045236 004737 054216  ;:66$:
12 045242 104412  JSR  PC,$TKINT      ;INITIALIZE THE TTY KEYBOARD
13 045244 012700 001230  RDOCT
14 045250 012701 000026  MOV  #RHDB,R0      ;GET STARTING ADDRESS OF REGISTERS
15 045254 012737 046034 000004  MOV  #22,R1        ;NUMBER OF REGISTERS
16 045262 021637 001240  MOV  #ADTIMO,4     ;SET UP TO CHECK NEW ADDRESS
17 045266 001407  CMP  (SP),RHCS1    ;NEW CSR?
18 045270 005776 000000  BEQ  1$           ;NO-SKIP NEXT
19 045274 163716 001240  TST  @ (SP)       ;ACCESS THE NEW ADDRESS
20 045300 061620  SUB  RHCS1,(SP)    ;GET THE ADDRESS OFFSET
21 045302 005301  ADD  (SP),(R0)+    ;AND PLUG IT IN
22 045304 001375  DEC  R1           ;DONE ALL OF THEM YET?
23 045306 0045314  BNE  2$          ;NOT YET, SO DO MORE
   045306 104401 045314  ;:1$:
   045312 000416  TYPE  ,69$          ;::TYPE ASCIZ STRING
   045350  ;:69$: .ASCIZ <CRLF>/PRESENT VECTOR ADDRESS IS /
24 045350 013746 001226  ;:68$:
25 045354 104402  MOV  RPVEC,-(SP)   ;GET READY TO TYPE OLD VECTOR ADDRESS
26 045356 104401 045364  TYPOC
   045362 000437  TYPE  ,71$          ;::TYPE ASCIZ STRING
   ;:71$: .ASCIZ <CRLF>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY 'CR' /
27 045462 104412  ;:70$:
28 045464 012637 001226  RDOCT
29 045470 104401 045476  MOV  (SP)+,RPVEC  ;SETUP VECTOR ADDRESS
   045474 000413  TYPE  ,73$          ;::TYPE ASCIZ STRING
   ;:73$: .ASCIZ <CRLF>/NEW BASE ADDRESS IS /
30 045524 013746 001240  ;:72$:
31 045530 104402  MOV  RHCS1,-(SP)
32 045532 104401 045540  TYPOC
   045536 000414  TYPE  ,75$          ;::TYPE ASCIZ STRING
   ;:75$: .ASCIZ <CRLF>/NEW VECTOR ADDRESS IS /
33 045570 013746 001226  ;:74$:
34 045574 104402  MOV  RPVEC,-(SP)
35 045576 104401 045604  TYPOC
   045602 000416  TYPE  ,77$          ;::TYPE ASCIZ STRING
   ;:77$: .ASCIZ <CRLF>/UNTIL PROGRAM :S RELOADED./
   045640  ;:76$:

```



```

36 045640 104401 045646      TYPE      79$      ::TYPE ASCIZ STRING
   045644 000402      BR        78$      ::GET OVER THE ASCIZ
   ::79$: .ASCIZ <CRLF>/ /
   78$:
37 045652 104401 045660      TYPE      81$      ::TYPE ASCIZ STRING
   045656 000424      BR        80$      ::GET OVER THE ASCIZ
   ::81$: .ASCIZ <CRLF>/UNLESS HALTED AND MANUALLY RESTARTED,/
   80$:
38 045730 104401 045736      TYPE      83$      ::TYPE ASCIZ STRING
   045734 000425      BR        82$      ::GET OVER THE ASCIZ
   ::83$: .ASCIZ <CRLF>/PROGRAM WILL AUTOMATICALLY RESTART FROM /
   82$:
39 046010 012746 000200      MOV      #200,-(SP)
40 046014 104402
41 046016 104401 046024      TYPE      85$      ::TYPE ASCIZ STRING
   046022 000402      BR        84$      ::GET OVER THE ASCIZ
   ::85$: .ASCIZ <CRLF>/ /
   84$:
42 046030 000137 004310      JMP      BEGIN      ;OK, NOW START OVER WITH NEW ADDRESS
43
44 046034
   046034 104401 046042      ADTIMO: TYPE      65$      ::TYPE ASCIZ STRING
   046040 000423      BR        64$      ::GET OVER THE ASCIZ
   ::65$: .ASCIZ <CRLF><377>/SPECIFIED ADDRESS DID NOT RESPOND. /
   64$:
45 046110 022626
46 046112 000137 045104      CMP      (SP)+,(SP)+ ;RESTORE THE STACK
   JMP      BASECH    ;AND DO IT AGAIN.
47
48 ;THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
49 ;THIS LOOPS HERE FOR EVER
50 ;TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
51 ;WITH WHAT IS REALY THERE
52
53 046116 000000      ERUNIT: 0      ;UNIT UNDER MANUAL TEST
54
55 046120 004737 041460      ERSTART:JSR    PC,CLDISK ;SET GENERAL REG.
56 046124 013712 046116      MOV      ERUNIT,@R2    ;SELECT UNIT
57 046130 005714      1$:      TST      @R4        ;TEST RHER1
58 046132 032712 010000      BIT      #NED,@R2     ;TEST NED
59 046136 001401      BEQ      2$          ;BRANCH IF GOOD
60 046140 000773      BR       1$          ;NED NOT SET
61 046142 000772      BR       1$          ;NED SET

```

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

```
.SBTTL DISK SIMULATION
:*****
:IN A WRITE HEADL AND DATA COMMAND FILL THE FOLLOWING
:   WCLY=WITH CYLINDER TO BE ON DISK
:   WSECTR=WITH SECTOR AND TRACK TO BE ON DISK
:   WKEY1= WITH KEY1 TO BE ON DISK
:   WKEY2= WITH KEY2 TO BE ON DISK
:   FNWORD= NO OF DATA WORDS TO BE WRITTEN ON DISK
:CALL
:   JSR   PC,COMWHD
:
:IN A WRITE DATA COMMAND FILL THE FOLLOWING
:   CYL=WITH CYLINDER TO BE FOUND ON DISK
:   SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
:   KEY1= WITH KEY1 TO BE FOUND ON DISK
:   KEY2= WITH KEY2 TO BE FOUND ON DISK
:   X= 1 MUST BE ONE
:   NOWORD= WITH NUMBER OF DATA WORDS TO BE WRITTEN
:CALL
:   JSR   PC,COMHD
:
:IN A READ HEADER AND DATA COMMAND FILL THE FOLLOWING
:   CYL= WITH CYLINDER TO BE FOUND ON DISK
:   SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
:   KEY1= WITH KEY1 TO BE FOUND ON DISK
:   KEY2=WITH KEY2 TO BE FOUND ON DISK
:   DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
:   X=0 MUST BE ZERO
:CALL
:   JSR   PC,COMHD
:
:IN A READ DATA COMMAND FILL THE FOLLOWING
:   CYL= WITH CYLINDER TO BE FOUND ON DISK
:   SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
:   KEY1= WITH KEY1 TO BE FOUND ON DISK
:   KEY2=WITH KEY2 TO BE FOUND ON DISK
:   DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
:   X=0 MUST BE ZERO
:CALL
:   JSR   PC,COMHD
:
:*****
:WRITE DATA COMMAND
:OR READ COMMAND I.E DATA ONLY OR HEADER AND DATA
:*****
:THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES
:IT ISSURE DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
:GO BIT
:IT THEN CALL THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
```



```

1
2
3
4
5
6
7 046410 014400
8 046412 000000
9 046414 000000
10 046416 000000
11 046420 000000
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26 046422 000000
27
28 046424 000000
29 046426 000000
30
31
32
33
34 046430 012137 046412
35 046434 012137 046414
36 046440 012137 046416
37 046444 012137 046420
38 046450 012137 047220
39 046454 010146
40 046456 013700 001260
41 046462 012705 000002
42 046466 012710 000001
43 046472 052710 000010
44 046476 052710 000002
45 046502 042710 000012
46 046506 000404
47 046510 012710 000013
48 046514 042710 000012
49 046520 012702 000007
50 046524 052710 000002
51 046530 042710 000002
52 046534 005302
53 046536 001372
54 046540 005305
55 046542 001362
56 046544 012702 000022
57 046550 005037 047216
    
```

```

;*****
;THE DISK SECTOR IS DEVIDED AS FOLLOWS
;19 WORDS OF 0, ONE WORD 144000
;THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE
RSYNC: 14400
RCYL: 0
RSETR: 0
RKEY1: 0
RKEY2: 0

;5 WORDS OF 0 ONE WORD 144000
;THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE
;THESE ARE DCL GENERATED

;THERE ARE 256 WORDS OF DATA
;THERE ARE 2 WORDS FOR ECC GENERATED BY DCL
;15 WORDS OF 0 FOR DATA GAP AND TOLERANCE GAP
;*****
;*****
;READ DISK HEADER
;*****
NOSYNC: 0 ;FORCED HEADER ERROR = -1
TY: 0 ;NORMAL = 0
ERWORD: 0 ;ERROR TYPE NO.
;ERROR WORD NO.

RDHEAD: MOV (R1)+,RCYL ;STORE CYLINDER ADDRESS
MOV (R1)+,RSETR ;STORE SECTOR AND TRACK ADDRESS
MOV (R1)+,RKEY1 ;STORE KEY1
MOV (R1)+,RKEY2 ;STORE KEY2
MOV (R1)+,COMPA ;STORE COMPARE OR NOT
MOV R1,-(SP) ;PUSH R1 ON STACK
MOV RHM,R0 ;R0 CONTAINS MAINTANENCE REG.
MOV #2,R5 ;R5 IS A COUNTER FOR WORDS
MOV #DMD,@R0 ;DIAG. MODE
BIS #MSTCK,@R0 ;SET SECTOR FOR FIRST WORD
BIS #MCLK,@R0 ;SET CLOCK FOR FIRST WORD
BIC #MSTCK!MCLK,@R0 ;RESET SECTOR AND CLOCK
BR 2$ ;BRANCH OVER GIVING SECTOR FOR FIRST TIME
1$: MOV #MSTCK!MCLK!DMD,@R0 ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
BIC #MSTCK!MCLK,@R0 ;RESET SECTOR, CLOCK
2$: MOV #7,R2 ;R2 IS A COUNTER FOR BYTES
3$: BIS #MCLK,@R0 ;SET CLOCK
BIC #MCLK,@R0 ;RESET CLOCK
DEC R2 ;BYTE COUNTER
BNE 3$ ;BRANCH IF BYTE NOT COMPLETE
DEC R5 ;WORD COUNTER
BNE 1$ ;BRANCH IF WORD NOT COMPLETE
56 MOV #18.,R2 ;NO OF WORDS OF ZEROS
57 CLR WORD ;READ 0
    
```

58	046554	004737	047222		JSR	PC,READ		:GO TO READ
59	046560	005302			DEC	R2		:COUNT
60	046562	001372			BNE	4\$		
61	046564	013737	046410	047216	MOV	RSYNC,WORD		:SYNC. WORD
62	046572	004737	047222		JSR	PC,READ		
63	046576	032710	001000		BIT	#DTSY,ARO		:SYNC. BYTE DETECTED?
64	046602	001012			BNE	5\$:BRANCH IF SYNC DETECTED
65	046604	012737	000001	046426	MOV	#1,ERWORD		:ERROR WORD NO
66	046612	013737	046410	001124	MOV	RSYNC,\$GDDAT		:SYNC WORD
67	046620	012737	104002	046320	MOV	#104002,SSYN		:INSERT 'ERROR 2' IN SSYN
68	046626	000571			BR	13\$:BRANCH OUT
69	046630	013737	046412	047216	5\$:	MOV	RCYL,WORD	:SETUP CYLINDER
70	046636	004737	047222		JSR	PC,READ		:READ
71	046642	013737	046414	047216	MOV	RSETR,WORD		:SETUP SECTOR/TRACK
72	046650	004737	047222		JSR	PC,READ		:READ
73	046654	013737	046416	047216	MOV	RKEY1,WORD		:SETUP KEY1
74	046662	004737	047222		JSR	PC,READ		:READ
75	046666	013737	046420	047216	MOV	RKEY2,WORD		:SETUP KEY2
76	046674	004737	047222		JSR	PC,READ		:READ
77	046700	013737	050206	047216	MOV	WCRC,WORD		:SETUP CRC
78	046706	004737	047222		JSR	PC,READ		:READ
79	046712	005737	001426		TST	TESDTE		:IS THIS A DRIVE TIMING ERROR
80	046716	001135			BNE	13\$:BRANCH OUT IF YES
81	046720	005737	047220		TST	COMPA		:IS THIS A READ OR WRITE COMMAND
82	046724	001472			BEQ	11\$		
83	046726	012705	050210		MOV	#HEGAP,R5		:POINTER FOR HEADER GAP
84	046732	012702	000005		MOV	#5,R2		:NO OF WORDS OF ZEROS
85	046736	012737	000006	046426	6\$:	MOV	#6,ERWORD	:ERROR WORD NO SET
86	046744	004737	047454		JSR	PC,WRITE		:FOR HEADER GAP
87	046750	005737	047452		TST	WWORD		:TEST WRITTEN WORD
88	046754	001413			BEQ	7\$:BRANCH IF GOOD THAT IS 0
89	046756	160237	046426		SUB	R2,ERWORD		:WORD NO IN ERROR
90	046762	005037	001124		CLR	\$GDDAT		:GOOD WORD SHOULD BE 0
91	046766	013737	047452	001126	MOV	WWORD,\$BDDAT		:BAD DATA
92	046774	012737	104003	046322	MOV	#104003,HEDGAP		: 'ERROR 2' GOES IN HEDGAP
93	047002	000503			BR	13\$:BRANCH OUT
94	047004	013725	047452		7\$:	MOV	WWORD,(R5)+	:SAVE HEADER GAP
95	047010	005302			DEC	R2		
96	047012	001351			BNE	6\$		
97	047014	004737	047454		JSR	PC,WRITE		:WRITE HEADER (DATA) GAP SYNC
98	047020	023737	046410	047452	CMP	RSYNC,WWORD		
99	047026	001426			BEQ	10\$		
100	047030	005737	046422		TST	NOSYNC		:IS THIS FORCED HEADER ERROR COMMAND
101								:IF YES NOSYNC=-1 THEN WRITE OR READ
102								:IS SHUT OFF SO BRANCH OUT
103								:IF NO NOSYNC=0 THEN CONTINUE
104	047034	001406			BEQ	14\$:BRANCH IF TRUE ERROR
105	047036	005737	047452		TST	WWORD		
106	047042	001420			BEQ	10\$:BRANCH IF GOOD
107	047044	005037	001124		CLR	\$GDDAT		:IT SHOULD BE ZERO
108	047050	000403			BR	15\$:BRANCH TO TYPE ERROR
109	047052	013737	046410	001124	14\$:	MOV	RSYNC,\$GDDAT	:GOOD DATA
110	047060	013737	047452	001126	15\$:	MOV	WWORD,\$BDDAT	:BAD DATA
111	047066	012737	000006	046426		MOV	#6,ERWORD	
112	047074	012737	104003	046324		MOV	#104003,HEDSYN	
113	047102	000443			BR	13\$:BRANCH OUT
114	047104	013725	047452		10\$:	MOV	WWORD,(R5)+	:SAVE DATA SYNC.

```

115 047110 000440          BR      13$
116
117          ;READ COMMAND START FROM HERE
118
119 047112 012702 000005    11$:  MOV    #5,R2
120 047116 005037 047216    12$:  CLR    WORD
121 047122 004737 047222    JSR    PC,READ      ;READ HEADER GAP
122 047126 005302          DEC    R2          ;IS 5 HEADER GAP ZEROS COMPLETE
123 047130 001372          BNE    12$        ;IF NOT BRANCH
124 047132 013737 046410 047216  MOV    RSYNC,WORD  ;SYNC WORD
125 047140 004737 047222    JSR    PC,READ      ;READ HEADER (DATA) SYNC)
126 047144 005737 046422    TST    NOSYNC
127 047150 001404          BEQ    16$        ;IF NOT ERROR COMMAND BRANCH
128 047152 032710 001000    BIT    #DTSY,@R0   ;SYNC. DETECTED
129 047156 001415          BEQ    13$        ;IF ZERO BRANCH OUT
130 047160 000403          BR     17$        ;IF NOT ZERO BRANCH TO ERROR
131 047162 032710 001000    16$:  BIT    #DTSY,@R0   ;SYNC. DETECTED?
132 047166 001011          BNE    13$        ;BRANCH IF YES
133 047170 012737 000006 046426 17$:  MOV    #6,ERWORD   ;ERROR WORD NO.
134 047176 013737 046410 001124  MOV    RSYNC,$GDDAT;SYNC WORD
135 047204 012737 104002 046324  MOV    #104002,HEDSYN
136 047212          13$:
    047212 012601          MOV    (SP)+,R1    ;:POP STACK INTO R1
137 047214 000201          RTS    R1
    
```

```

1
2
3
4
5
6 047216 000000
7 047220 000000
8
9 047222
10 047222 010246
11 047224 012705 000002
12 047230 012710 000001
13 047234 006037 047216
14 047240 103002
15 047242 052710 000020
16 047246 012702 000007
17 047252 052710 000012
18 047256 005737 044160
19 047262 001411
20 047264 032710 000020
21 047270 001404
22 047272 012737 177777 044152
23 047300 000402
24 047302 005037 044152
25 047306 012746 000001
26 047312 006037 047216
27 047316 103002
28 047320 012716 000021
29 047324 012610
30 047326 005737 044160
31 047332 001404
32 047334 005237 044172
33 047340 004737 044210
34 047344 052710 000002
35 047350 005737 044160
36 047354 001411
37 047356 032710 000020
38 047362 001404
39 047364 012737 177777 044152
40 047372 000402
41 047374 005037 044152
42 047400 012746 000001
43 047404 006037 047216
44 047410 103002
45 047412 012716 000021
46 047416 012610
47 047420 005737 044160
48 047424 001404
49 047426 005237 044172
50 047432 004737 044210
51 047436 005302
52 047440 001341
53 047442 005305
54 047444 001300
55 047446 012602
56 047450 000207

*****
:READ ONE WORD IN 'WORD'
*****
WORD: 0
COMPA: 0

READ:
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV #2,R5 ;:WORD COUNTER
MOV #DMD,@R0 ;:SET DIAG. MODE
ROR WORD ;:CHECKING IF THERE IS A ONE
BCC 1$ ;:IF NO ONE BRANCH
BIS #MRD,@R0 ;:SET BIT 4 IF DATA HAS ONE
1$: MOV #7,R2 ;:BYTE COUNTER
BIS #MSTCK!MCLK,@R0 ;:SET CLOCK,DATA IF ANY, SECTOR
TST TSECCG ;:IS THIS BIT TO GENERATE AND TEST ECC
BEQ 6$ ;:BRANCH IF NO
BIT #MRD,@R0 ;:IS DATA BIT A ONE
BEQ 5$ ;:BRANCH IF DATA BIT IS 0
MOV #-1,ECDATA ;:ECC DATA BIT IS A ONE
BR 6$ ;:BRANCH
5$: CLR ECDATA ;:ECC DATA BIT IS A 0
6$: MOV #DMD,-(SP) ;:KEEP ONLY DIAG. MODE
ROR WORD ;:CHECKING IF THERE IS A ONE
BCC 2$ ;:IF NO ONE BRANCH
MOV #MRD!DMD,(SP) ;:KEEP DATA AND DIAG. MODE
2$: MOV (SP)+,@R0 ;:PUT IN DATA,RESET CLOCK, SECTOR
TST TSECCG ;:IS ECC TO BE GENERATED FOR THIS BIT
BEQ 3$ ;:BRANCH IF NO
INC DATENV ;:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
3$: JSR PC,ECTEST ;:GO TO GENERATE AND TEST ECC
BIS #MCLK,@R0 ;:SET CLOCK
TST TSECCG ;:IS THIS BIT TO GENERATE ECC
BEQ 8$ ;:BRANCH IF NO
BIT #MRD,@R0 ;:IS DATA BIT A ONE
BEQ 7$ ;:BRANCH IF DATA BIT IS = 0
MOV #-1,ECDATA ;:ECC DATA BIT IS A ONE
BR 8$ ;:BRANCH
7$: CLR ECDATA ;:ECC DATA BIT IS = 0
8$: MOV #DMD,-(SP) ;:KEEP DIAG. MODE
ROR WORD ;:CHECKING IF THERE IS A ONE
BCC 4$ ;:BRANCH IF NO ONE
MOV #MRD!DMD,(SP) ;:KEEP DIAG. MODE AND DATA
4$: MOV (SP)+,@R0 ;:SET DATA, DIAG. MODE, CLEAR CLOCK
TST TSECCG ;:IS THIS BIT TO GENERATE ECC
BEQ 9$ ;:BRANCH IF NO
INC DATENV ;:NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
9$: JSR PC,ECTEST ;:GO TO GENERATE AND TEST ECC
DEC R2 ;:BYTE COUNTER
BNE 3$ ;:BRANCH IF ONE BYTE NOT COMPLETE
DEC R5 ;:WORD COUNTER
BNE 1$ ;:BRANCH IF ONE WORD NOT COMPLETE
MOV (SP)+,R2 ;:POP STACK INTO R2
RTS PC

```



```

1
2
3
4
5
6 047452 000000
7
8 047454
   047454 010046
   047456 010246
   047460 010346
   047462 010546
9 047464 012705 000002
10 047470 012710 000001
11
12 047474 012702 000007
13 047500 012710 000013
14 047504 032710 000040
15 047510 001406
16 047512 012737 177777 044152
17 047520 000261
18 047522 006003
19 047524 000404
20 047526 005037 044152
21 047532 000241
22 047534 006003
23 047536 012710 000001
24 047542 005737 044160
25 047546 001404
26 047550 005237 044172
27 047554 004737 044210
28 047560 052710 000002
29 047564 032710 000040
30 047570 001406
31 047572 012737 177777 044152
32 047600 000261
33 047602 006003
34 047604 000404
35 047606 005037 044152
36 047612 000241
37 047614 006003
38 047616 012710 000001
39 047622 005737 044160
40 047626 001404
41 047630 005237 044172
42 047634 004737 044210
43 047640 005302
44 047642 001346
45 047644 005305
46 047646 001312
47
48 047650 010337 047452
49
50 047654 012605
   047656 012603
   047660 012602
   047662 012600

*****
WRITE ONE WORD WHICH COMES BACK IN 'WORD'
*****

WORD: 0

WRITE:
MOV R0,-(SP)      ;;PUSH R0 ON STACK
MOV R2,-(SP)      ;;PUSH R2 ON STACK
MOV R3,-(SP)      ;;PUSH R3 ON STACK
MOV R5,-(SP)      ;;PUSH R5 ON STACK
MOV #2,R5         ;WORD COUNTER
MOV #1,@R0        ;SET DIAG. MODE
MOV #7,R2         ;R0 HAS RHMR ADDRESS IN IT
                    ;BYTE COUNTER
1$: MOV #MSTCK!MCLK!DMD,@R0 ;SET SECTOR AND CLOCK
    MOV #MWR,@R0    ;CHECK WRITE BIT IN MAINT. REG.
    BIT #MWR,@R0    ;CHECK WRITE BIT IN MAINT. REG.
    BEQ 2$          ;BRANCH IF ZERO
    MOV #-1,ECDATA ;ECC DATA BIT IS A ONE
    SEC            ;SET CARRY
    ROR R3         ;MOVE 1 FORWARD
    BR 3$
2$: CLR ECDATA    ;ECC DATA BIT IS = 0
    CLC           ;CLEAR CARRY
    ROR R3       ;MOVE 0 FOR WORD
3$: MOV #DMD,@R0 ;CLEAR SECTOR AND CLOCK
    TST TSECCG  ;IS THIS BIT TO GENERATE ECC
    BEQ 4$     ;BRANCH IF NO
    INC DATENV ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
    JSR PC,ECTEST ;GO TO GENERATE AND TEST ECC
4$: BIS #MCLK,@R0 ;SET CLOCK IN RHMR
    BIT #MWR,@R0 ;CHECK WRITE BIT IN RHMR
    BEQ 5$     ;BRANCH IF ZERO
    MOV #-1,ECDATA ;ECC DATA BIT IS A ONE
    SEC       ;SET CARRY
    ROR R3   ;MOVE 1 FOR WORD
    BR 6$
5$: CLR ECDATA ;ECC DATA BIT IS ZERO
    CLC   ;CLEAR CARRY
    ROR R3 ;MOVE 0 FOR WORD
6$: MOV #DMD,@R0 ;CLEAR CLOCK
    TST TSECCG ;IS THIS BIT TO GENERATE ECC
    BEQ 7$   ;BRANCH IF NO
    INC DATENV ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
    JSR PC,ECTEST ;GO TO GENERATE AND TEST ECC
7$: DEC R2 ;COUNT FOR BYTE END
    BNE 4$ ;IF NOT BYTE END BRANCH
    DEC R5 ;COUNT FOR WORD END
    BNE 1$ ;IF NOT WORD END BRANCH

MOV R3,WORD ;STORE THE WORD

MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R0 ;;POP STACK INTO R0
    
```

51 047664 000207

RTS PC

C
S

```

1
2
3
4
5
6
7 047666 000000
8 047670 000400
9 047672 000000
10 047674
11 047674 011137 047666
12 047700 012102
13 047702 012137 047220
14
15 047706 010046
    047710 010146
    047712 010246
    047714 010346
    047716 010446
16
17 047720 012701 000016
18 047724 012703 051232
19 047730 012723 177777
20 047734 005301
21 047736 001374
22 047740 013700 001260
23 047744 013746 047670
24 047750 163716 047666
25 047754 011637 047672
26 047760 012604
27 047762 005737 001424
28 047766 001403
29 047770 012737 177777 044160
30 047776 012703 050224
31 050002 004737 047454
32 050006 013723 047452
33 050012 005302
34 050014 001372
35 050016 005704
36 050020 001406
37 050022 004737 047454
38 050026 013723 047452
39 050032 005304
40 050034 001372
41 050036 005037 044160
42 050042 012701 000002
43 050046 004737 047454
44 050052 013723 047452
45 050056 005301
46 050060 001372
47 050062 004737 047454
48 050066 013723 047452
49 050072 012701 000016
50 050076 004737 047454
51 050102 013723 047452
52 050106 005301
53 050110 001372
    
```

 :WRITE DATA - PUT DATA INTO 'DISK' AREA FROM 'WORD'
 :ONE WORD AT A TIME
 :*****

COUNTD: 0
 FORMAT: 256.
 ZWORDS: 0
 WRDATA:

```

MOV (R1),COUNTD ;STORE NO. OF WORDS TO BE WRITTEN
MOV (R1)+,R2 ;SAME IN R2
MOV (R1)+,COMPA ;COMPARE OR NOT

MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK

MOV #14.,R1 ;NO. OF TOLERANCE GAP WORDS
MOV #TOLGAP,R3 ;START OF TOLERANCE GAP TABLE
1$: MOV #-1,(R3)+ ;MAKE IT 177777
DEC R1 ;IS 14 COMPLETED
BNE 1$ ;IF NO BRANCH
MOV RMMR,R0 ;R0 CONTAINS MAINTANENCE REG.
MOV FORMAT,-(SP)
SUB COUNTD,(SP)
MOV (SP),ZWORDS ;NO. OF ZERO WORDS TO BE WRITTEN
MOV (SP)+,R4
TST TSECC ;IS THIS AN ECC TEST ?
BEQ 7$ ;BRANCH IF NO
MOV #-1,TSECCG ;THESE BITS ARE TO GENERATE ECC
7$: MOV #DISK,R3 ;ADDRESS THE 'DISK' AREA
2$: JSR PC,WRITE ;WRITE INTO 'WORD'
MOV WORD,(R3)+ ;STORE ON SIMULATED DISK
DEC R2 ;COUNT DOWN
BNE 2$ ;CONTINUE IF ALL WORDS NOT WRITTEN
TST R4 ;ANY ZEROS TO BE WRITTEN ?
BEQ 4$ ;BRANCH IF NONE TO BE WRITTEN
3$: JSR PC,WRITE ;WRITE ZEROS INTO 'WORD'
MOV WORD,(R3)+ ;STORE INTO 'DISK'
DEC R4
BNE 3$
4$: CLR TSECCG ;NO MORE ECC TO BE GENERATED
MOV #2,R1
5$: JSR PC,WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
MOV WORD,(R3)+ ;STORE ON WEEC1 AND WEEC2
DEC R1
BNE 5$
6$: JSR PC,WRITE ;WRITE DATA GAP INTO 'WORD'
MOV WORD,(R3)+ ;STORE INTO 'DISK'
MOV #14.,R1
6$: JSR PC,WRITE ;WRITE TOLERANCE GAP ZEROS
MOV WORD,(R3)+ ;STORE INTO 'DISK'
DEC R1
BNE 6$
    
```


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

```
*****  
:WRITE HEADER AND DATA  
*****  
:THIS IS THE SIMULATED DISK  
:ONLY ONE SECTOR OF SPACE IS ALLOWED  
*****  
SECGAP: .BLKW 19. ;SECTOR GAP 38 BYTES OF 0  
WSSYNC: .BLKW 1 ;SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE  
HEADER: .BLKW 4 ;HEADER = CYL, SECTOR/TRACK, KEY1, KEY2  
WCRC: .BLKW 1 ;CRC  
HEGAP: .BLKW 5 ;HEADER GAP 10 BYTES OF 0  
HDWSYN: .BLKW 1 ;HEADER GAP 1 BYTE OF 0 ONE SYNC. BYTE  
SILOTB: ;USED IN SILO TEST AS SILO TABLE  
DISK: .BLKW 256. ;DATA SPACE  
WECC1: .BLKW 1 ;ECC1  
WECC2: .BLKW 1 ;ECC2  
DTAGAP: .BLKW 1 ;DATA GAP 2 BYTES OF 0  
TOLGAP: .BLKW 14. ;TOLERANCE GAP 28 BYTES OF 0  
*****  
:WRITE HEADER AND DATA  
*****  
:THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES  
:IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE  
:'GO' BIT  
:IT THEN CALL THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER  
:SUBROUTINES. THESE ARE:  
:SEARCH  
:WRHEAD  
:WRDATA  
RNCTR1: .WORD 0 ;'RUN' LINE STALL COUNTER  
COMWHD: MOV (SP),PCJSR ;SAVE PC OF JSR + 4  
SUB #4,PCJSR ;SAVE PC OF JSR  
MOV R0,-(SP) ;PUSH R0 ON STACK  
MOV R1,-(SP) ;PUSH R1 ON STACK  
MOV R2,-(SP) ;PUSH R2 ON STACK  
MOV R3,-(SP) ;PUSH R3 ON STACK  
MOV R4,-(SP) ;PUSH R4 ON STACK  
MOV R5,-(SP) ;PUSH R5 ON STACK  
44 051316 012777 000001 127734 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE  
45 051324 052777 000004 127726 BIS #MINX,@RHMR ;SET DIAGNOSTIC INDEX  
46 051332 042777 000004 127720 BIC #MINX,@RHMR ;CLEAR IT  
47 051340 052777 000001 127672 BIS #GO,@RHCS1 ;SET 'GO' BIT & STALL 'TILL 'RUN'  
49 051346 012737 000113 051266 RNWAT1: MOV #75.,RNCTR1 ;LOAD STALL COUNTER = APPROX. 450US  
50 ;FOR 11/50 CPU WITH CORE MEMORY  
51 051354 005337 051266 1$: DEC RNCTR1 ;COUNT DOWN 1 TIME  
52 051360 001. '5 BNE 1$ ;CONTINUE UNTIL 0
```

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 051362 013746 051446      MOV      WSECTR, -(SP)      ;GET DESIRED SECTOR/TRACK
55 051366 042716 177740      BIC      #177740, (SP)     ;MAKE ONLY SECTOR
56 051372 012637 051402      MOV      (SP)+, WTRK      ;SAVE SECTOR
57
58 051376 004137 052354      JSR      R1, SEARCH      ;ISSUE SECTOR CLOCKS
59 051402 000000      WTRK:   .WORD 0          ;SECTOR NO.
60
61 051404 012701 000240      MOV      #+NOP, R1       ;GOING TO MOVE NOPS
62 051410 010137 051456      MOV      R1, SEGPEN     ;NOP INTO SEGAP
63 051414 010137 051460      MOV      R1, FSYNER     ;NOP INTO FSYNER
64 051420 010137 051462      MOV      R1, ERHEAD     ;NOP INTO ERHEAD
65 051424 010137 051464      MOV      R1, ERCRC     ;NOP INTO ERCRC
66 051430 010137 051466      MOV      R1, ERHDGP    ;NOP INTO ERHDGP
67 051434 010137 051470      MOV      R1, HDESYN    ;NOP INTO HDESYN
68
69 051440 004137 051540      JSR      R1, WRHEAD     ;WRITE THE HEADER
70 051444 000000      WCYL:   0              ;CYLINDER
71 051446 000000      WSECTR: 0              ;SECTOR AND TRACK
72 051450 000000      WKEY1:  0              ;KEY1
73 051452 000000      WKEY2:  0              ;KEY2
74 051454 000000      GCRC:   0              ;GOOD CRC
75
76      ;DUMMY ERROR CALL LOCATIONS FOR THE WRITE HEADER OPERATION
77
78 051456 000240      SEGPEN: NOP           ;IF 'ERROR 6' INSERTED BY
79      ;WRHEAD SUBROUTINE THEN
80      ;SECTOR GAP GOING ON DISK
81      ;IS NOT RIGHT.
82
83      ;WORD NO. CONTAINS WHICH
84      ;WORD IS WRONG, THAT IS
85      ;FIRST OF TENTH OR WHAT EVER NO.
86      ;BAD WORD IS GOING ON DISK
87
88 051460 000240      FSYNER: NOP           ;IF 'ERROR 6' INSERTED BY
89      ;WRHEAD SUBROUTINE THEN
90      ;THE LAST 0 BYTE OF SECTOR
91      ;GAP, OR FIRST SYNC. BYTE
92      ;AFTER SECTOR GAP IS IN
93      ;ERROR.
94
95      ;WORD NO. CONTAINS 20
96      ;RIGHT BYTE IS SECTOR GAP
97      ;LEFT BYTE IS SYNC. BYTE
98      ;BAD WORD IS WHAT IS GOING ON
99      ;DISK.
100
101 051462 000240      ERHEAD: NOP          ;IF 'ERROR 6' INSERTED BY
102      ;WRHEAD SUBROUTINE THEN
103      ;HEADER GOING ON DISK
104      ;IS WRONG.
105
106      ;WORD NO 1 = CYLINDER NO
107      ;WORD NO 2 = SECTOR/TRACK
108      ;WORD NO 3 = KEY1
109      ;WORD NO 4 = KEY2

```

```

110                                     ;BAD WORD IS WHAT IS GOING ON
111                                     ;DISK
112
113 051464 000240      ERCRC: NOP      ;IF 'ERROR 6' INSERTED BY
114                                     ;WRHEAD SUBROUTINE THEN CRC WRITTEN
115                                     ;ON DISK IS IN ERROR.
116
117                                     ;GOOD DATA IS WHAT SHOULD BE ON DISK
118                                     ;BAD DATA IS WHAT IS GOING ON DISK
119                                     ;WORD NO IS 5.
120
121 051466 000240      ERHDGP: NOP     ;IF 'ERROR 6' INSERTED BY
122                                     ;WRHEAD SUBROUTINE THEN HEADER
123                                     ;GAP GOING ON DISK IS WRONG.
124
125                                     ;WORD NO. GIVES WHICH OF
126                                     ;THE HEADER GAP WORDS
127                                     ;ARE WRONG. FOR EXAMPLE:
128
129                                     ;WORD NO 1 = FIRST HEADER
130                                     ;          GAP WORD
131                                     ;BAD WORD IS WHAT IS GOING ON DISK
132
133 051470 000240      HDESYN: NOP     ;IF 'ERROR 6' INSERTED BY
134                                     ;WRHEAD SUBROUTINE THEN LAST
135                                     ;HEADER GAP BYTE OR HEADER
136                                     ;SYNC BYTE GOING ON DISK IS WRONG.
137
138                                     ;WORD NO = 5
139                                     ;BAD DATA IS WHAT IS GOING
140                                     ;ON DISK, RIGHT BYTE IS HEADER
141                                     ;GAP 0 BYTE, LEFT BYTE IS HEADER
142                                     ;GAP SYNC.
143
144 051472 005737 001406      TST      ERFLGS
145 051476 001004      BNE      FOUT
146
147 051500 004137 047674      FNWORD: JSR      R1,WRDATA
148 051504 000000      .WORD    0      ;WRITE THE DATA
149 051506 000000      .WORD    0      ;FORMAT COMMAND NO. OF DATA
150
151 FOUT:
151 051510 012605      MOV      (SP)+,R5  ;;POP STACK INTO R5
151 051512 012604      MOV      (SP)+,R4  ;;POP STACK INTO R4
151 051514 012603      MOV      (SP)+,R3  ;;POP STACK INTO R3
151 051516 012602      MOV      (SP)+,R2  ;;POP STACK INTO R2
151 051520 012601      MOV      (SP)+,R1  ;;POP STACK INTO R1
151 051522 012600      MOV      (SP)+,R0  ;;POP STACK INTO R0
152
153 051524 000207      RTS      PC

```

```

1
2
3
4
5
6
7
8
9
10
11
12 051526 000000
13 051530 000000
14 051532 000000
15 051534 000000
16 051536 000000
17
18 051540 012137 051526
19 051544 012137 051530
20 051550 012137 051532
21 051554 012137 051534
22 051560 012137 051536
23 051564 010146
24 051566 012701 050126
25 051572 013700 001260
26 051576 012710 000001
27 051602 012705 000002
28 051606 052710 000010
29 051612 012710 000013
30 051616 032710 000040
31 051622 001403
32 051624 000261
33 051626 006003
34 051630 000402
35 051632 000241
36 051634 006003
37 051636 012710 000001
38 051642 012702 000007
39 051646 052710 000002
40 051652 032710 000040
41 051656 001403
42 051660 000261
43 051662 006003
44 051664 000402
45 051666 000241
46 051670 006003
47 051672 012710 000001
48 051676 005302
49 051700 001362
50 051702 005305
51 051704 001342
52
53 051706 010321
54 051710 005703
55 051712 001414
56 051714 012737 000001 046426
57 051722 005037 001124

;*****
;WRITE HEADER
;*****

;R0 = MAINT.REG.
;R1 = SIMULATED DISK
;R2 = BYTE COUNT
;R3 = WRITE WORD
;R5 = WORD COUNT

SCYL: 0
SSECTR: 0
SKEY1: 0
SKEY2: 0
SCRC: 0

WRHEAD: MOV (R1)+,SCYL
        MOV (R1)+,SSECTR
        MOV (R1)+,SKEY1
        MOV (R1)+,SKEY2
        MOV (R1)+,SCRC
        MOV R1,-(SP)
        MOV #SECGAP,R1
        MOV RHMR,R0
        MOV #DMD,@R0
        MOV #2,R5
        BIS #MSTCK,@R0
1$: MOV #MSTCK!MCLK!DMD,@R0
   BIT #MWR,@R0
   BEQ 2$
   SEC
   ROR R3
   BR 3$
2$: CLC
   ROR R3
3$: MOV #DMD,@R0
   MOV #7,R2
4$: BIS #MCLK,@R0
   BIT #MWR,@R0
   BEQ 5$
   SEC
   ROR R3
   BR 6$
5$: CLC
   ROR R3
6$: MOV #DMD,@R0
   DEC R2
   BNE 4$
   DEC R5
   BNE 1$

;PUSH R1 ON STACK
;SIMULATED DISK INDICATOR
;R0 NOW HAS MAINT. REG. ADDR.
;SET DIAG. MODE IN RHMR
;WORD COUNTER
;SET SECTOR FOR FIRST BYTE
;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
;CHECK WRITE BIT IN MAINT. REG.

;SET CARRY
;MOVE ONE FORWARD
;
;CLEAR CARRY
;MOVE ZERO FORWARD
;CLEAR CLOCK, SECTOR
;BYTE COUNTER
;SET CLOCK
;CHECK WRITE BIT IN MAINT.REG.
;BRANCH IF ZERO
;SET CARRY
;MOVE ONE FORWARD

;SET DIAG. MODE AGAIN IN RHMR

;CONTINUE

MOV R3,(R1)+
TST R3
BEQ 7$
MOV #1,ERWORD
CLR $GDDAT
    
```



```

58 051726 010337 001126          MOV    R3,$BDDAT
59 051732 012737 104006 051456    MOV    #104006,SEGP
60 051740 000137 052346          JMP    17$                ;BRANCH OUT
61
62 051744 012702 000022          MOV    #18.,R2           ;COUNT NO. OF SECTOR GAP
63 051750 012737 000024 046426 7$:  MOV    #20.,ERWORD      ;COUNT TO GIVE ERROR WORD
64 051756 004737 047454          JSR    PC,WRITE         ;WRITE SECTOR GAP
65 051762 013721 047452          MOV    WWORD,(R1)+      ;STORE SECTOR GAP WORD
66 051766 001413          BEQ    11$
67 051770 160237 046426          SUB    R2,ERWORD        ;IF NOT GET ERROR WORD NO.
68 051774 005037 001124          CLR    $GDDAT          ;GOOD WORD
69 052000 013737 047452 001126    MOV    WWORD,$BDDAT;BAD WORD
70 052006 012737 104006 051456    MOV    #104006,SEGP;STORE 'ERROR 6' IN SEGP
71 052014 000554          BR     17$                ;BRANCH OUT
72
73 052016 005302          11$:  DEC    R2                ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
74 052020 001353          BNE    10$                ;IF NOT DO SO
75
76          ;AT THIS POINT THE SECTOR FOUND FLOP SHOULD
77          ;BE HIGH.  SO THAT THE HEADER SYNC BYTE CAN BE GIVEN
78
79          ;HOWEVER IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
80          ;IS ABORTED - HEADER SYNC BYTE IS NOT GIVEN
81
82 052022 005737 001426          TST    TESDTE          ;IS THIS A DRIVE TIMING ERROR
83 052026 001147          BNE    17$                ;BRANCH OUT IF YES
84 052030 004737 047454          JSR    PC,WRITE         ;WRITE ONE SECTOR GAP 0 BYTE
85          ;AND ONE SYNC. BYTE = 230
86 052034 013711 047452          MOV    WWORD,(R1)      ;SAVE 0 BYTE AND SYNC BYTE
87 052040 023721 046410          CMP    RSYNC,(R1)+     ;IF SYNC. BYTE RIGHT
88 052044 001414          BEQ    12$                ;IF YES BRANCH
89 052046 012737 000024 046426    MOV    #20.,ERWORD      ;IF NOT GET READY FOR ERROR
90 052054 013737 046410 001124    MOV    RSYNC,$GDDAT;GOOD WORD
91 052062 014137 001126          MOV    -(R1),$BDDAT     ;BAD WORD
92 052066 012737 104006 051460    MOV    #104006,FSYNER;INSERT 'ERROR 6' IN FSYNER
93 052074 000524          BR     17$                ;BRANCH OUT
94
95 052076 012702 000004          12$:  MOV    #4,R2            ;FOUR HEADER WORDS
96 052102 012703 051526          MOV    #SCYL,R3        ;POINTER FOR HEADER TABLE
97 052106 012737 000005 046426 13$:  MOV    #5,ERWORD        ;ERROR WORD NO SET
98 052114 004737 047454          JSR    PC,WRITE         ;WRITE 4 HEADER WORDS
99 052120 013711 047452          MOV    WWORD,(R1)      ;STORE WRITTEN WORD
100 052124 022321          CMP    (R3)+,(R1)+     ;IS IT RIGHT?
101 052126 001412          BEQ    14$                ;IF GOOD CONTINUE
102          ;IF NOT GET READY FOR PRINT
103 052130 160237 046426          SUB    R2,ERWORD        ;WORD NO
104 052134 014337 001124          MOV    -(R3),$GDDAT     ;GOOD DATA
105 052140 014137 001126          MOV    -(R1),$BDDAT     ;BAD DATA
106 052144 012737 104006 051462    MOV    #104006,ERHEAD;INSERT 'ERROR 6'
107 052152 000475          BR     17$                ;BRANCH OUT
108
109 052154 005302          14$:  DEC    R2                ;ARE 4 HEADER WORDS DONE?
110 052156 001353          BNE    13$                ;IF NOT DO THEM
111 052160 004737 047454          JSR    PC,WRITE         ;WRITE CRC
112 052164 013711 047452          MOV    WWORD,(R1)      ;STORE CRC
113 052170 022137 051454          CMP    (R1)+,GCRC      ;COMPARE GOOD CRC
114 052174 001414          BEQ    20$                ;BRANCH IF GOOD
    
```

115	052176	014137	001126			MOV	-(R1), \$BDDATA	:BAD CRC WRITTEN
116	052202	013737	051454	001124		MOV	GCRC, \$GDDAT	:GOOD CRC
117	052210	012737	000005	046426		MOV	#5, ERWORD	:ERROR WORD NO
118	052216	012737	104006	051464		MOV	#104006, ERCRC	:INSERT ERROR 6
119	052224	000450				BR	17\$:EXIT
120								
121	052226	012702	000005		20\$:	MOV	#5, R2	:NO OF HEADER GAP
122	052232	012737	000006	046426	15\$:	MOV	#6, ERWORD	:ERROR WORD NO SET
123	052240	004737	047454			JSR	PC, WRITE	:WRITE HEADER GAP
124	052244	013721	047452			MOV	WWORD, (R1)+	:STORE
125	052250	001412				BEQ	16\$:IF GOOD BRANCH
126	052252	160237	046426			SUB	R2, ERWORD	:ERROR WORD NO
127	052256	005037	001124			CLR	\$GDDAT ;GOOD DATA	
128	052262	014137	001126			MOV	-(R1), \$BDDAT	:BAD DATA
129	052266	012737	104006	051466		MOV	#104006, ERHDGP; STORE 'ERROR 6'	
130	052274	000424				BR	17\$:BRANCH OUT
131								
132	052276	005302			16\$:	DEC	R2	:ARE 5 HEADER GAP ZEROS DONE
133	052300	001354				BNE	15\$:IF NOT BRANCH
134	052302	004737	047454			JSR	PC, WRITE	
135	052306	013711	047452			MOV	WWORD, (R1)	
136	052312	023721	046410			CMP	RSYNC, (R1)+	
137	052316	001413				BEQ	17\$:EXIT
138	052320	012737	000005	046426		MOV	#5, ERWORD	
139	052326	014137	001126			MOV	-(R1), \$BDDAT	
140	052332	013737	046410	001124		MOV	RSYNC, \$GDDAT	
141	052340	012737	104006	051470		MOV	#104006, HDESYN	
142								
143	052346				17\$:	MOV	(SP)+, R1	::POP STACK INTO R1
144	052346	012601						
145	052350	000201				RTS	R1	

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

```

:*****
:SEARCH SECTOR
:*****
    
```

```

:      R0=RMMR ADDRESS
:      R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
:      R2=CLOCK COUNT (PER BYTE)
:      R3=SECTOR COUNTER FROM R1
:      R5=BYTES PER WORD COUNT
:BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
:SECTOR PULSE IN CASE IT IS SET
:AT BEGINNING OF EACH SECTOR ONE SECTOR CLOCK HAS TO RISE
:BEFORE CLOCK THEN EVERY EIGHT CLOCKS ONE SECTOR CLOCK IS
:IDENTICAL WITH CLOCK
:NUMBERING THE SECTOR CLOCKS AS FOLLOWS
:THE SECTOR CLOCK UNDER INDEX - 0
:THE NEXT - 1
:THE NEXT - 2
:ETC.
:THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER - 608
:THE NEXT SECTOR THEN HAS 608 SECTOR CLOCKS
:THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS
:AND SO ON
    
```

```

SECTR: 0 ;SECTOR SEARCHED FOR
SEARCH: MOV (R1)+,SECTR ;SAVE SECTOR SEARCHED FOR
        MOV R0,-(SP) ;PUSH R0 ON STACK
        MOV R1,-(SP) ;PUSH R1 ON STACK
        MOV R2,-(SP) ;PUSH R2 ON STACK
        MOV R3,-(SP) ;PUSH R3 ON STACK
        MOV R4,-(SP) ;PUSH R4 ON STACK
        MOV R5,-(SP) ;PUSH R5 ON STACK
        MOV RMMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADR.
        MOV SECTR,R3 ;SECTOR COUNTER
        MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
        BIS #MSTCK,@R0 ;SET SECTOR CLOCK
        BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
        BIS #MSTCK,@R0 ;SET SECTOR CLOCK
        BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK
        ;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
        ;RESETTING SECTOR PULSE
        ;IN CASE IT STARTS SET
        BIS #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
        MOV #DMD,@R0 ;RESET INDEX AND SECTOR CLOCK
        TST R3 ;IF SECTOR REQUIRED JUMP OUT
        BEQ 7$ ;BRANCH OF SECTOR ZERO REQUIRED
    
```

```

;NOW THE 304 WORDS WILL START
;FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH
;BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN
;FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK
;WILL BE IDENTICAL WITH ONE CLOCK
;ONE WORD ONLY
    
```

```

53
54 052444 012702 000010      1$:  MOV      #8.,R2      ;BYTE COUNTER
55 052450 012705 000002      MOV      #2,R5      ;BYTES PER WORD
56 052454 052710 000010      BIS      #MSTCK,@R0 ;SET SECTOR CLOCK
57 052460 052710 000002      BIS      #MCLK,@R0  ;SET CLOCK
58 052464 000402              BR       3$         ;BRANCH TO CLEAR SECTOR AND CLOCK
59 052466 052710 000012      2$:  BIS      #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
60 052472 042710 000012      3$:  BIC      #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
61 052476 052710 000002      8$:  BIS      #MCLK,@R0   ;SET CLOCK
62 052502 042710 000002      BIC      #MCLK,@R0   ;CLEAR CLOCK
63 052506 005302              DEC      R2         ;BYTE COUNTER
64 052510 001372              BNE     8$         ;BRANCH IF BYTE NOT COMPLETE
65 052512 012702 000007      MOV      #7,R2      ;SETUP FOR SECOND BYTE
66 052516 005305              DEC      R5         ;IS WORD COMPLETE?
67 052520 001362              BNE     2$         ;BRANCH IF NOT COMPLETE
68                          ;'0 GIVE SECTOR CLOCK AND CLOCK
69
70                          ;NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL
71
72 052522 012701 000457      MOV      #303.,R1   ;WORDS PER SECTOR COUNTER
73 052526 012705 000002      4$:  MOV      #2,R5      ;BYTES PER WORD COUNTER
74 052532 012702 000007      5$:  MOV      #7,R2      ;BYTE COUNTER (CLOCK COUNTER)
75 052536 052710 000012      BIS      #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
76 052542 042710 000012      BIC      #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
77 052546 052710 000002      6$:  BIS      #MCLK,@R0   ;SET CLOCK
78 052552 042710 000002      BIC      #MCLK,@R0   ;RESET CLOCK
79 052556 005302              DEC      R2         ;IS BYTE COMPLETE?
80 052560 001372              BNE     6$         ;BRANCH IF NOT COMPLETE
81 052562 005305              DEC      R5         ;IS WORD COMPLETE?
82 052564 001362              BNE     5$         ;BRANCH IF NOT
83 052566 005301              DEC      R1         ;IS SECTOR COMPLETE
84 052570 001356              BNE     4$         ;BRANCH IF NOT
85 052572 052710 000010      BIS      #MSTCK,@R0  ;SET SECTOR
86 052576 042710 000010      BIC      #MSTCK,@R0  ;CLEAR SECTOR
87 052602 005303              DEC      R3         ;IS REQUIRED NO OF SECTORS COMPLETE
88 052604 001317              BNE     1$         ;BRANCH IF NOT
89
90 052606      7$:  MOV      (SP)+,R5      ;;POP STACK INTO R5
    052606 012605      MOV      (SP)+,R4      ;;POP STACK INTO R4
    052610 012604      MOV      (SP)+,R3      ;;POP STACK INTO R3
    052612 012603      MOV      (SP)+,R2      ;;POP STACK INTO R2
    052614 012602      MOV      (SP)+,R1      ;;POP STACK INTO R1
    052616 012601      MOV      (SP)+,R0      ;;POP STACK INTO R0
    052620 012600      MOV      (SP)+,R0
    052622 000201      RTS      R1
91
92
93                          ;:*****
94                          ;:READ ONE SECTOR OF DATA
95                          ;:*****
96
97 052624 000000      RNO:    0           ;NO. OF WORDS READ
98 052626 000000      RCOM:   0           ;EXTRA STORAGE
99
100 052630 012137 052624      REDATA: MOV      (R1)+,RNO ;SAVE NO. OF WORDS ONLY FOR INFORMATION
101 052634 012137 052626      MOV      (R1)+,RCOM ;EXTRA WORD ONLY FOR INFORMATION
102 052640 010146              MOV      R1,-(SP)    ;;PUSH R1 ON STACK
103 052642 005737 001424      TST     TSECC      ;IS THIS AN ECC TEST
    
```

```

104 052646 001403          BEQ      1$          ;BRANCH IF NO
105 052650 012737 177777 044160  MOV     #-1,TSECCG    ;THESE BITS ARE TO GENERATE ECC
106 052656 012702 000402 1$:    MOV     #258.,R2      ;256 WORDS PER SECTOR
107                                     ;PLUS 2 ECC WORDS
108 052662 012703 050224          MOV     #DISK,R3      ;POINTE TO DISK SIMULATION
109 052666 012337 047216 2$:    MOV     (R3)+,WORD    ;READY TO READ CONTENTS
110 052672 004737 047222          JSR     PC,READ       ;READ
111 052676 005302          DEC     R2            ;IS 256 WORDS DONE?
112 052700 001372          BNE     2$           ;IF NOT BRANCH
113 052702 005737 001424          TST     TSECC        ;IS THIS AN ECC TEST
114 052706 001012          BNE     4$           ;BRANCH OUT IF YES
115 052710 005037 044160          CLR     TSECCG       ;NO MORE ECC BITS ARE TO BE GENERATED
116 052714 012702 000017          MOV     #15.,R2      ;ONE DATA GAP, 14 TOLERANCE GAP
117 052720 012337 047216 3$:    MOV     (R3)+,WORD    ;READY TO READ CONTENTS OF WORD
118 052724 004737 047222          JSR     PC,READ ;READ
119 052730 005302          DEC     R2            ;COUNT
120 052732 001372          BNE     3$           ;BRANCH IF 14 NOT DONE
121 052734 4$:
122 052734 012601          MOV     (SP)+,R1     ;:POP STACK INTO R1
123 052736 000201          RTS      R1          ;:RETURN
124 052740          RPVECT:
125 052740 104401 052746          TYPE   ,65$         ;:TYPE ASCIZ STRING
126 052744 000420          BR     ,64$         ;:GET OVER THE ASCIZ
127 053006          ;:65$: .ASCIZ /UNEXPECTED RP INTERRUPT @ PC = /
128 053006          ;:64$:
129 053006 104402          TYPOC          ;:TYPE FROM PC
130 053010 012777 052740 126210  MOV     #RPVECT,@RPVEC ;:RESTORE TRAP VECTOR
131 053016 000002          RTI          ;:CHANGE TO CONTINUE
    
```



```

053206 000002 RTI
053210 012637 000004 2001$: MOV (SP)+,ERRVEC ;;RESTORE CONTENTS OF ERROR VECTOR

053214 022737 177777 056026 2002$: CMP #-1,CPSAVE ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
053222 001430 BEQ 2003$ ;;BRANCH IF SO
053224 032737 000001 056026 BIT #BIT00,CPSAVE ;;SEE IF THE POWER MONITOR BIT IS ON
053232 001424 BEQ 2003$ ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
053234 042737 000001 177766 BIC #BIT00,177766 ;;CLEAR THE BIT FOUND TO BE SET
053242 013746 001140 MOV SWR,-(SP) ;;SAVE SWR ADDRESS
053246 017646 000000 MOV @ (SP),-(SP) ;;SAVE SWR VALUE
053252 012737 000176 001140 MOV #176,SWR ;;GET SOFTWARE SWR ADDRESS
053260 011677 125654 MOV (SP),@SWR ;;GET CURRENT SWR VALUE
053264 042777 001000 125646 BIC #BIT09,@SWR ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
053272 104177 EMT 177 ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
053274 012676 000000 MOV (SP)+,@(SP) ;;RESTORE SWR TO ORIGINAL VALUE
053300 012637 001140 MOV (SP)+,SWR ;;RESTORE SWR ADDRESS
053304 2003$:
053304 123737 001115 001103 CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
053312 101015 BHI 3$ ;;BR IF NO
053314 032777 001000 125616 BIT #BIT09,@SWR ;;LOOP ON ERROR?
053322 001404 BEQ 4$ ;;BR IF NO
053324 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
053332 000443 BR $OVER
053334 105037 001103 4$: CLR $ERFLG ;;ZERO THE ERROR FLAG
053340 005037 001212 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
053344 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
053346 032777 004000 125564 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
053354 001011 BNE 1$ ;;BR IF YES
053356 005737 001100 TST $PASS ;;IF FIRST PASS OF PROGRAM
053362 001406 BEQ 1$ ;; INH'BIT ITERATIONS
053364 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
053370 023737 001212 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
053376 002021 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
053400 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
053406 013737 053456 001212 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
053414 105237 001102 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
053420 011637 001106 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
053424 011637 001110 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
053430 005037 001214 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
053434 112737 000001 001115 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
053442 013777 001102 125472 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
053450 013716 001106 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
053454 000002 RTI ;;FIXES PS
053456 000004 $MXCNT: 4 ;;MAX. NUMBER OF ITERATIONS
    
```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 *REPLACED WITH SPACES.
 *CALL:

* MOV NUM,-(SP) ;:PUT THE BINARY NUMBER ON THE STACK
 * TYPDS ;:GO TO THE ROUTINE

053460				\$TYPDS:	MOV	R0,-(SP)	::PUSH R0 ON STACK
053460	010046				MOV	R1,-(SP)	::PUSH R1 ON STACK
053462	010146				MOV	R2,-(SP)	::PUSH R2 ON STACK
053464	010246				MOV	R3,-(SP)	::PUSH R3 ON STACK
053466	010346				MOV	R5,-(SP)	::PUSH R5 ON STACK
053470	010546				MOV	#20200,-(SP)	::SET BLANK SWITCH AND SIGN
053472	012746	020200			MOV	20(SP),R5	::GET THE INPUT NUMBER
053476	016605	000020			BPL	1\$::BR IF INPUT IS POS.
053502	100004				NEG	R5	::MAKE THE BINARY NUMBER POS.
053504	005405				MOVB	#'-,1(SP)	::MAKE THE ASCII NUMBER NEG.
053506	112766	000055	000001		CLR	R0	::ZERO THE CONSTANTS INDEX
053514	005000			1\$:	MOV	#\$DBLK,R3	::SETUP THE OUTPUT POINTER
053516	012703	053674			MOVB	#',(R3)+	::SET THE FIRST CHARACTER TO A BLANK
053522	112723	000040			CLR	R2	::CLEAR THE BCD NUMBER
053526	005002			2\$:	MOV	\$DTBL(R0),R1	::GET THE CONSTANT
053530	016001	053664			SUB	R1,R5	::FORM THIS BCD DIGIT
053534	160105			3\$:	BLT	4\$::BR IF DONE
053536	002402				INC	R2	::INCREASE THE BCD DIGIT BY 1
053540	005202				BR	3\$	
053542	000774				ADD	R1,R5	::ADD BACK THE CONSTANT
053544	060105			4\$:	TST	R2	::CHECK IF BCD DIGIT=0
053546	005702				BNE	5\$::FALL THROUGH IF 0
053550	001002				TSTB	(SP)	::STILL DOING LEADING 0'S?
053552	105716				BMI	7\$::BR IF YES
053554	100407				ASLB	(SP)	::MSD?
053556	106316			5\$:	BCC	6\$::BR IF NO
053560	103003				MOVB	1(SP),-1(R3)	::YES--SET THE SIGN
053562	116663	000001	177777		BIS	#'0,R2	::MAKE THE BCD DIGIT ASCII
053570	052702	000060		6\$:	BIS	#',R2	::MAKE IT A SPACE IF NOT ALREADY A DIGIT
053574	052702	000040		7\$:	MOVB	R2,(R3)+	::PUT THIS CHARACTER IN THE OUTPUT BUFFER
053600	110223				TST	(R0)+	::JUST INCREMENTING
053602	005720				CMP	R0,#10	::CHECK THE TABLE INDEX
053604	020027	000010			BLT	2\$::GO DO THE NEXT DIGIT
053610	002746				BGT	8\$::GO TO EXIT
053612	003002				MOV	R5,R2	::GET THE LSD
053614	010502				BR	6\$::GO CHANGE TO ASCII
053616	000764				TSTB	(SP)+	::WAS THE LSD THE FIRST NON-ZERO?
053620	105726			8\$:	BPL	9\$::BR IF NO
053622	100003				MOVB	-1(SP),-2(R3)	::YES--SET THE SIGN FOR TYPING
053624	116663	177777	177776		CLRB	(R3)	::SET THE TERMINATOR
053632	105013			9\$:	MOV	(SP)+,R5	::POP STACK INTO R5
053634	012605				MOV	(SP)+,R3	::POP STACK INTO R3
053636	012603				MOV	(SP)+,R2	::POP STACK INTO R2
053640	012602				MOV	(SP)+,R1	::POP STACK INTO R1
053642	012601						

053644	012600			MOV	(SP)+,F0	::POP STACK INTO R0
053646	104401	053674		TYPE	\$DBLK	::NOW TYPE THE NUMBER
053652	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
053660	012616			MOV	(SP)+,(CP)	
053662	000002			RTI		::RETURN TO USER
053664	023420		\$DTBL:	10000.		
053666	001750			1000.		
053670	000144			100.		
053672	000012			10.		
053674			\$DBLK:	.BLKW	4	

.SBTTL TYPE ROUTINE

```

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

```

```

*CALL:
*1) USING A TRAP INSTRUCTION
*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
*      TYPE
*      MESADR

```

```

053704 105737 001157 $TYPE: TSTB $TPFLG      ;; IS THERE A TERMINAL?
053710 100002      BPL      1$      ;; BR IF YES
053712 000000      HALT      ;; HALT HERE IF NO TERMINAL
053714 000407      BR      3$      ;; LEAVE
053716 010046      1$: MOV      RO,-(SP)  ;; SAVE RO
053720 017600 000002 2$: MOV      @2(SP),RO  ;; GET ADDRESS OF ASCIZ STRING
053724 112046      MOVB     (RO)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
053726 001005      BNE      4$      ;; BR IF IT ISN'T THE TERMINATOR
053730 005726      TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
053732 012600      60$: MOV      (SP)+,RO  ;; RESTORE RO
053734 062716 000002 3$: ADD      #2,(SP)      ;; ADJUST RETURN PC
053740 000002      RTI      ;; RETURN
053742 122716 000011 4$: CMPB     #HT,(SP)      ;; BRANCH IF <HT>
053746 001430      BEQ      8$      ;;
053750 122716 000200      CMPB     #CRLF,(SP)  ;; BRANCH IF NOT <CRLF>
053754 001006      BNE      5$      ;;
053756 005726      TST      (SP)+      ;; POP <CR><LF> EQUIV
053760 104401      TYPE      ;; TYPE A CR AND LF
053762 001223      $CRLF
053764 105037 054172      CLRB     $CHARCNT  ;; CLEAR CHARACTER COUNT
053770 000755      BR      2$      ;; GET NEXT CHARACTER
053772 004737 054054 5$: JSR      PC,$TYPEC  ;; GO TYPE THIS CHARACTER
053776 123726 001156 6$: CMPB     $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
054002 001350      BNE      2$      ;; IF NO GO GET NEXT CHAR.
054004 013746 001154      MOV      $NULL,-(SP)  ;; GET # OF FILLER CHARS. NEEDED
                                ;; AND THE NULL CHAR.
054010 105366 000001 7$: DECB     1(SP)      ;; DOES A NULL NEED TO BE TYPED?
054014 002770      BLT      6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK
054016 004737 054054      JSR      PC,$TYPEC  ;; GO TYPE A NULL
054022 105337 054172      DECB     $CHARCNT  ;; DO NOT COUNT AS A COUNT
054026 000770      BR      7$      ;; LOOP

```

;HORIZONTAL TAB PROCESSOR

```

054030 112716 000040 8$: MOVB     #' ,(SP)      ;; REPLACE TAB WITH SPACE
054034 004737 054054 9$: JSR      PC,$TYPEC  ;; TYPE A SPACE
054040 132737 000007 054172 BITB     #7,$CHARCNT  ;; BRANCH IF NOT AT
054046 001372      BNE      9$      ;; TAB STOP
054050 005726      TST      (SP)+      ;; POP SPACE OFF STACK
054052 000724      BR      2$      ;; GET NEXT CHARACTER

```

```

054054                                $TYPEC:
054054 105777 125064                TSTB  @STKS      ;;CHAR IN KYBD BUFFER?
054060 100022                        BPL    10$      ;;BR IF NOT
054062 017746 125060                MOV    @STKB,-(SP) ;;GET CHAR
054066 042716 177600                BIC    #177600,(SP) ;;STRIP EXTRANEIOUS BITS
054072 122716 000023                CMPB  #$XOFF,(SP) ;;WAS CHAR XOFF
054076 001012                        BNE    102$    ;;BR IF NOT
054100                                101$:
054100 105777 125040                TSTB  @STKS      ;;WAIT FOR CHAR
054104 100375                        BPL    101$
054106 117716 125034                MOVB  @STKB,(SP) ;;GET CHAR
054112 042716 177600                BIC    #177600,(SP) ;;STRIP IT
054116 122716 000021                CMPB  #$XON,(SP) ;;WAS IT XON?
054122 001366                        BNE    101$    ;;BR IF NOT
054124                                102$:
054124 005726                        TST   (SP)+    ;;FIX STACK
054126                                10$:
054126 105777 125016                TSTB  @STPS      ;;WAIT UNTIL PRINTER IS READY
054132 100375                        BPL    10$
054134 116677 000002 125010          MOVB  2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
054142 122766 000015 000002          CMPB  #CR,2(SP)  ;;IS CHARACTER A CARRIAGE RETURN?
054150 001003                        BNE    1$      ;;BRANCH IF NO
054152 105037 054172                CLMB  $CHARCNT  ;;YES--CLEAR CHARACTER COUNT
054156 000406                        BR     $TYPEX   ;;EXIT
054160 122766 000012 000002 1$:    CMPB  #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
054166 001402                        BEQ   $TYPEX   ;;BRANCH IF YES
054170 105227                        INCB  (PC)+    ;;COUNT THE CHARACTER
054172 000000                        $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
054174 000207                        $TYPEX: RTS   PC
  
```

.SBTTL TTY INPUT ROUTINE

```

*****
ENABL  LSB
054176 000000 $TKCNT: .WORD 0          ;;NUMBER OF ITEMS IN QUEUE
054200 000000 $TKQIN: .WORD 0          ;;INPUT POINTER
054202 000000 $TKQOUT: .WORD 0         ;;OUTPUT POINTER
054204 054215 $TKQSRV: .BLKB 9.      ;;TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
;*CALL:
;*      JSR      PC,$TKINT
;*      RETURN
054216 005037 054176 $TKINT: CLR $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
054222 012737 054204 054200 MOV # $TKQSRV,$TKQIN      ;;MOVE THE STARTING ADDRESS OF THE
054230 013737 054200 054202 MOV $TKQIN,$TKQOUT      ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
054236 012737 054266 000060 MOV # $TKSRV,@TKVEC      ;;INITIALIZE THE KEYBOARD VECTOR
054244 012737 000200 000062 MOV #200,@TKVEC+2      ;;'BR' LEVEL 4
054252 005777 124670 TST @TKB                ;;CLEAR DONE FLAG
054256 012777 000100 124660 MOV #100,@STKS          ;;ENABLE TTY KEYBOARD INTERRUPT
054264 000207 RTS PC                ;;RETURN TO CALLER

;*TK SERVICE ROUTINE
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;*IT IN THE QUEUE.
;*IF THE CHARACTER IS A 'CONTROL-C' (^C) $TKINT IS CALLED AND
;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (OPERSEL)
054266 117746 124654 $TKSRV: MOVB @TKB, -(SP)      ;;PICKUP THE CHARACTER
054272 042716 177600 BIC #^C177, (SP)          ;;STRIP THE JUNK
054276 021627 000021 CMP (SP),#$XON           ;;IS IT A RANDOM XON?
054302 001002 BNE 30$                ;;BRANCH IF NO
054304 005726 TST (SP)+              ;;CLEAN RANDOM XON OFF STACK
054306 000002 RTI                ;;RETURN
054310 30$:
054310 021627 000003 CMP (SP),#3              ;;IS IT A CONTROL C?
054314 001007 BNE 1$                ;;BRANCH IF NO
054316 104401 055267 TYPE , $CNTLC          ;;TYPE A CONTROL-C (^C)
054322 004737 054216 JSR PC,$TKINT          ;;INIT THE KEYBOARD
054326 005726 TST (SP)+              ;;CLEAN UP STACK
054330 000137 040310 JMP OPERSEL           ;;CONTROL C RESTART
054334 021627 000007 1$: CMP (SP),#7          ;;IS IT A CONTROL G?
054340 001004 BNE 2$                ;;BRANCH IF NO
054342 022737 000176 001140 CMP #SWREG,SWR          ;;IS SOFT-SWR SELECTED?
054350 001500 BEQ 6$                ;;GO TO SWR CHANGE
054352 2$:
054352 022737 000011 054176 CMP #9, $TKCNT          ;;IS THE QUEUE FULL?
054360 001004 BNE 3$                ;;BRANCH IF NO
054362 104401 001216 TYPE , $BELL          ;;RING THE TTY BELL
  
```

```
054366 005726          TST      (SP)+      ;;CLEAN CHARACTER OFF OF STACK
054370 000451          BR       5$        ;;EXIT
054372 021627 000023 3$:  CMP      (SP),#23  ;;IS IT A CONTROL-S?
054376 001021          BNE     32$        ;;BRANCH IF NO
054400 005077 124540          CLR     @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
054404 005726          TST      (SP)+      ;;CLEAN CHAR OFF STACK
054406 105777 124532 31$:  TSTB   @STKS      ;;WAIT FOR A CHAR
054412 100375          BPL     31$        ;;LOOP UNTIL ITS THERE
054414 117746 124526          MOVB   @STKB,-(SP) ;;GET THE CHARACTER
054420 042716 177600          BIC     #^C177,(SP) ;;MAKE IT 7-BIT ASCII
054424 022627 000021          CMP     (SP)+,#21  ;;IS IT A CONTROL-Q?
054430 001366          BNE     31$        ;;BRANCH IF NO
054432 012777 000100 124504  MOV     #100,@STKS  ;;REENABLE TTY KEYBOARD INTERRUPTS
054440 000002          RTI                    ;;RETURN
054442 005237 054176 32$:  INC     $TKCNT      ;;COUNT THIS CHARACTER
054446 021627 000140          CMP     (SP),#140  ;;IS IT UPPER CASE?
054452 002405          BLT     4$        ;;BRANCH IF YES
054454 021627 000175          CMP     (SP),#175  ;;IS IT A SPECIAL CHAR?
054460 003002          BGT     4$        ;;BRANCH IF YES
054462 042716 000040          BIC     #40,(SP)   ;;MAKE IT UPPER CASE
054466 112677 177506 4$:  MOVB   (SP)+,@$TKQIN ;;AND PUT IT IN QUEUE
054472 005237 054200          INC     $TKQIN     ;;UPDATE THE POINTER
054476 023727 054200 054215  CMP     $TKQIN,$$TKQEND ;;GO OFF THE END?
054504 001003          BNE     5$        ;;BRANCH IF NO
054506 012737 054204 054200  MOV     $$TKQSR,$$TKQIN ;;RESET THE POINTER
054514 000002 5$:  RTI                    ;;RETURN
```

*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
*CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```
054516 022737 000176 001140 $CKSWR: CMP     #SWREG,SWR  ;;IS THE SOFT-SWR SELECTED
054524 001124          BNE     15$        ;;EXIT IF NOT
054526 105777 124412          TSTB   @STKS      ;;IS A CHAR WAITING?
054532 100121          BPL     15$        ;;IF NOT, EXIT
054534 117746 124406          MOVB   @STKB,-(SP) ;;YES
054540 042716 177600          BIC     #^C177,(SP) ;;MAKE IT 7-BIT ASCII
054544 021627 000007          CMP     (SP),#7   ;;IS IT A CONTROL-G?
054550 001300          BNE     2$        ;;IF NOT, PUT IT IN THE TTY QUEUE
                                ;;AND EXIT
```

*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

```
054552 123727 001134 000001 6$:  CMPB   $AUTOB,#1   ;;ARE WE RUNNING IN AUTO-MODE?
054560 001674          BEQ     2$        ;;BRANCH IF YES
054562 005726          TST      (SP)+      ;;CLEAR CONTROL-G OFF STACK
054564 004737 054216          JSR     PC,$TKINT  ;;FLUSH THE TTY INPUT QUEUE
054570 005077 124350          CLR     @STKS      ;;DISABLE TTY KEYBOARD INTERRUPTS
054574 112737 000001 001135  MOVB   #1,$INTAG   ;;SET INTERRUPT MODE INDICATOR
```

```
054602 104401 055301          TYPE    , $CNTLG   ;;ECHO THE CONTROL-G (^G)
054606 104401 055306  $GTSWR: TYPE    , $MSWR   ;;TYPE CURRENT CONTENTS
054612 013746 000176          MOV     SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
054616 104402          TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```

054620 104401 055317          TYPE      ,SMNEW          ;;PROMPT FOR NEW SWR
054624 005046          19$: CLR      -(SP)          ;;CLEAR COUNTER
054626 005046          CLR      -(SP)          ;;THE NEW SWR
054630 105777 124310      7$: TSTB   @STKS          ;;CHAR THERE?
054634 100375          BPL      7$          ;;IF NOT TRY AGAIN

054636 117746 124304      MOVB   @STKB, -(SP)      ;;PICK UP CHAR
054642 042716 177600      BIC    #^C177, (SP)      ;;MAKE IT 7-BIT ASCII

054646 021627 000003      CMP     (SP), #3          ;;IS IT A CONTROL-C?
054652 001015          BNE    9$          ;;BRANCH IF NOT
054654 104401 055267      TYPE   ,SCNTLC          ;;YES, ECHO CONTROL-C (^C)
054660 062706 000006      ADD    #6, SP          ;;CLEAN UP STACK
054664 123727 001135 000001  CMPB   $INTAG, #1        ;;REENABLE TTY KEYBOARD INTERRUPTS?
054672 001003          BNE    8$          ;;BRANCH IF NO
054674 012777 000100 124242  MOV    #100, @STKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
054702 000137 040310      8$: JMP    OPERSEL      ;;CONTROL-C RESTART

054706 021627 000025      9$: CMP     (SP), #25        ;;IS IT A CONTROL-U?
054712 001005          BNE    10$         ;;BRANCH IF NOT
054714 104401 055274      TYPE   ,SCNTLU          ;;YES, ECHO CONTROL-U (^U)
054720 062706 000006      20$: ADD    #6, SP          ;;IGNORE PREVIOUS INPUT
054724 000737          BR     19$          ;;LET'S TRY IT AGAIN

054726 021627 000015      10$: CMP    (SP), #15        ;;IS IT A <CR>?
054732 001022          BNE    16$         ;;BRANCH IF NO
054734 005766 000004      TST    4(SP)          ;;YES, IS IT THE FIRST CHAR?
054740 001403          BEQ    11$          ;;BRANCH IF YES
054742 016677 000002 124170  MOV    2(SP), @SWR      ;;SAVE NEW SWR
054750 062706 000006      11$: ADD    #6, SP          ;;CLEAR UP STACK
054754 104401 001223      14$: TYPE   ,SCRLF          ;;ECHO <CR> AND <LF>
054760 123727 001135 000001  CMPB   $INTAG, #1        ;;RE-ENABLE TTY KBD INTERRUPTS?
054766 001003          BNE    15$         ;;BRANCH IF NOT
054770 012777 000100 124146  MOV    #100, @STKS      ;;RE-ENABLE TTY KBD INTERRUPTS
054776 000002          RTI                    ;;RETURN
055000 004737 054054      15$: JSR    PC, $TYPEPC      ;;ECHO CHAR
055004 021627 000060      16$: CMP    (SP), #60        ;;CHAR < 0?
055010 002420          BLT    18$         ;;BRANCH IF YES
055012 021627 000067      CMP    (SP), #67        ;;CHAR > 7?
055016 003015          BGT    18$         ;;BRANCH IF YES
055020 042726 000060      BIC    #60, (SP)+      ;;STRIP-OFF ASCII
055024 005766 000002      TST    2(SP)          ;;IS THIS THE FIRST CHAR
055030 001403          BEQ    17$         ;;BRANCH IF YES
055032 006316          ASL    (SP)          ;;NO, SHIFT PRESENT
055034 006316          ASL    (SP)          ;;CHAR OVER TO MAKE
055036 006316          ASL    (SP)          ;;ROOM FOR NEW ONE.
055040 005266 000002      17$: INC    2(SP)          ;;KEEP COUNT OF CHAR
055044 056616 177776      BIS    -2(SP), (SP)    ;;SET IN NEW CHAR
055050 000667          BR     7$          ;;GET THE NEXT ONE
055052 104401 001222      18$: TYPE   ,SQUES          ;;TYPE ?<CR><LF>
055056 000720          BR     20$         ;;SIMULATE CONTROL-U

.DSABL  LSB

```

;;*****

;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

;*CALL:

```

:*      RDCHR          ;;GET A CHARACTER FROM THE QUEUE
:*      RETURN HERE   ;;CHARACTER IS ON THE STACK
:*                               ;;WITH PARITY BIT STRIPPED OFF

```

```

055060 011646          SRDCHR: MOV      (SP),-(SP)    ;;PUSH DOWN THE PC AND
055062 016666 000004 000002  MOV      4(SP),2(SP)    ;;THE PS
055070 005066 000004          CLR      4(SP)      ;;GET READY FOR A CHARACTER
055074 005046          CLR      -(SP)      ;;PUT NEW PS ON STACK
055076 012746 055104          MOV      #64$,-(SP)    ;;PUT NEW PC ON STACK
055102 000002          RTI          ;;POP NEW PC AND PS
055104
055104 005737 054176 64$:   1$:   TST      $TKCNT    ;;WAIT ON A CHARACTER
055110 001775          BEQ      1$
055112 005337 054176          DEC      $TKCNT    ;;DECREMENT THE COUNTER
055116 117766 177060 000004  MOVB    @STKQOUT,4(SP)  ;;GET ONE CHARACTER
055124 005237 054202          INC      $TKQOUT    ;;UPDATE THE POINTER
055130 023727 054202 054215  CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
055136 001003          BNE     2$          ;;BRANCH IF NO
055140 012737 054204 054202  MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
055146 000002          RTI          ;;RETURN

```

;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY

;*CALL:

```

:*      RDLIN          ;;INPUT A STRING FROM THE TTY
:*      RETURN HERE   ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
:*                               ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

```

```

055150 010346          SRDLIN: MOV     R3, -(SP)    ;;SAVE R3
055152 012703 055256 1$:   MOV     #$TTYIN,R3    ;;GET ADDRESS
055156 022703 055267 2$:   CMP     #$TTYIN+9.,R3  ;;BUFFER FULL?
055162 101405          BLOS   4$          ;;BR IF YES
055164 104410          RDCHR          ;;GO READ ONE CHARACTER FROM THE TTY
055166 112613          MOVB   (SP)+,(R3)    ;;GET CHARACTER
055170 122713 000177 10$:  CMPB   #177,(R3)    ;;IS IT A RUBOUT
055174 001003          BNE   3$          ;;SKIP IF NOT
055176 104401 001222 4$:   TYPE   ,SQUES    ;;TYPE A '?'
055202 000763          BR     1$          ;;CLEAR THE BUFFER AND LOOP
055204 111337 055254 3$:   MOVB   (R3),9$    ;;ECHO THE CHARACTER
055210 104401 055254          TYPE   ,9$
055214 122723 000015          CMPB   #15,(R3)+    ;;CHECK FOR RETURN
055220 001356          BNE   2$          ;;LOOP IF NOT RETURN
055222 105063 177777          CLRB  -1(R3)      ;;CLEAR RETURN (THE 15)
055226 104401 001224          TYPE   ,SLF      ;;TYPE A LINE FEED
055232 012603          MOV   (SP)+,R3    ;;RESTORE R3
055234 011646          MOV   (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
055236 016666 000004 000002  MOV     4(SP),2(SP)    ;;FIRST ASCII CHARACTER ON IT
055244 012766 055256 000004  MOV     #$TTYIN,4(SP)
055252 000002          RTI          ;;RETURN
055254 000          9$:   .BYTE   0          ;;STORAGE FOR ASCII CHAR. TO TYPE
055255 000          .BYTE   0          ;;TERMINATOR
055256          .BLKB   9          ;;RESERVE 9 BYTES FOR TTY INPUT
055267 136 103 015 $CNTLC: .ASCIZ /^C/<15><12> ;;CONTROL 'C'
055274 136 125 015 $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
055301 136 107 015 $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'

```

055306	015	012	123	\$MSWR:	.ASCIZ	<15><12>/SWR = /
055317	040	040	116	\$MNEW:	.ASCIZ	/ NEW = /

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

```

:*****
:*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
:*CHANGE IT TO BINARY.
:*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
:*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
:*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
:*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
:*CALL:
:*      RDOCT          ;;READ AN OCTAL NUMBER
:*      RETURN HERE   ;;LOW ORDER BITS ARE ON TOP OF THE STACK
:*                  ;;HIGH ORDER BITS ARE IN $HIOCT
  
```

```

055330 011646          $RDOCT: MOV      (SP),-(SP)      ;;PROVIDE SPACE FOR THE
055332 016666 000004 000002 MOV      4(SP),2(SP)      ;;INPUT NUMBER
055340 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
055342 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
055344 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
055346 104411          1$:  RDLIN                    ;;READ AN ASCII LINE
055350 012600          MOV      (SP)+,R0          ;;GET ADDRESS OF 1ST CHARACTER
055352 010037 055456  MOV      R0,5$          ;;AND SAVE IT
055356 005001          CLR      R1          ;;CLEAR DATA WORD
055360 005002          CLR      R2
055362 112046          2$:  MOVB     (R0)+,-(SP)      ;;PICKUP THIS CHARACTER
055364 001420          BEQ      3$          ;;IF ZERO GET OUT
055366 122716 000060  CMPB     #'0,(SP)      ;;MAKE SURE THIS CHARACTER
055372 003026          BGT      4$          ;;IS AN OCTAL DIGIT
055374 122716 000067  CMPB     #'7,(SP)
055400 002423          BLT      4$
055402 006301          ASL      R1          ;;*2
055404 006102          ROL      R2
055406 006301          ASL      R1          ;;*4
055410 006102          ROL      R2
055412 006301          ASL      R1          ;;*8
055414 006102          ROL      R2
055416 042716 177770  BIC      #'C7,(SP)      ;;STRIP THE ASCII JUNK
055422 062601          ADD      (SP)+,R1      ;;ADD IN THIS DIGIT
055424 000756          BR       2$          ;;LOOP
055426 005726          3$:  TST      (SP)+          ;;CLEAN TERMINATOR FROM STACK
055430 010166 000012  MOV      R1,12(SP)      ;;SAVE THE RESULT
055434 010237 055466  MOV      R2,$HIOCT
055440 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
055442 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
055444 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
055446 000002          RTI
055450 005726          4$:  TST      (SP)+          ;;CLEAN PARTIAL FROM STACK
055452 105010          CLRB    (R0)          ;;SET A TERMINATOR
055454 104401          TYPE
055456 000000          5$:  .WORD   0          ;;TYPE UP THRU THE BAD CHAR.
055460 104401 001222  TYPE     $QUES      ;; '?' 'CR' & 'LF'
055464 000730          BR       1$          ;;TRY AGAIN
055466 000000          $HIOCT: .WORD   0      ;;HIGH ORDER BITS GO HERE
  
```

.SBTTL ERROR HANDLER ROUTINE

```

:*****
:*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
:*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
:*AND GO TO $ERRTYP ON ERROR
:*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:*SW15=1      HALT ON ERROR
:*SW13=1      INHIBIT ERROR TYPEOUTS
:*SW10=1      BELL ON ERROR
:*SW09=1      LOOP ON ERROR
:*CALL
:*          ERROR  N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER
    
```

```

055470 105037 056030 $ERROR: CLRB IBSAVE          ;;CLEAR THE ITEM BYTE SAVE LOCATION
055474 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
055476 012737 177777 001406 MOV      #-1,ERFLG$        ;;SET ERROR FLAG
055504          REGSA1:
055504 105237 001103 7$:      INCB $ERFLG          ;;SET THE ERROR FLAG
055510 001775          BEQ      7$          ;;DON'T LET THE FLAG GO TO ZERO
055512 013777 001102 123422 MOV      $TSTNM,@DISPLAY    ;;DISPLAY TEST NUMBER AND ERROR FLAG
055520 032777 002000 123412 BIT      #BIT10,@SWR        ;;BELL ON ERROR?
055526 001402          BEQ      1$          ;;NO - SKIP
055530 104401 001216          TYPE $BELL          ;;RING BELL
055534 005237 001112 1$:      INC      $ERTTL        ;;COUNT THE NUMBER OF ERRORS
055540 011637 001116          MOV      (SP),$ERRPC        ;;GET ADDRESS OF ERROR INSTRUCTION
055544 162737 000002 001116 SUB      #2,$ERRPC
055552 117737 123340 001114 MOVB    @$ERRPC,$ITEMB     ;;STRIP AND SAVE THE ERROR ITEM CODE
055560 032777 001000 123352 BIT      #BIT09,@SWR        ;;SEE IF LOOP ON ERROR IS SET
055566 001060          BNE      1004$        ;;BRANCH AROUND ROUTINE IF SO
055570 122737 000177 001114 CMPB    #177,$ITEMB        ;;SEE IF THIS IS THE POWER FAIL CALL
055576 001454          BEQ      1004$        ;;BRANCH AROUND ROUTINE IF IT IS
055600 105737 056030          TSTB IBSAVE          ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
055604 001047          BNE      1003$        ;;BRANCH IF SO
055606 022737 177777 056026 CMP      #-1,CPSAVE        ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
055614 001445          BEQ      1004$        ;;BRANCH IF SO
055616 013746 000004          MOV      ERRVEC,-(SP)        ;;SAVE CONTENTS OF ERROR VECTOR
055622 012737 055640 000004 MOV      #1000$,ERRVEC     ;;SETUP 'TRAP' RETURN ADDRESS
055630 013737 177766 056026 MOV      177766,CPSAVE     ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
055636 000406          BR      1001$
055640 012737 177777 056026 1000$: MOV      #-1,CPSAVE        ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
055646 012716 055654          MOV      #1001$, (SP)        ;;SETUP RETURN ADDRESS
055652 000002          RTI
055654 012637 000004          1001$: MOV      (SP)+,ERRVEC        ;;RESTORE CONTENTS OF ERROR VECTOR

055660 022737 177777 056026 1002$: CMP      #-1,CPSAVE        ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
055666 001420          BEQ      1004$        ;;BRANCH IF SO
055670 032737 000001 056026 BIT      #BIT00,CPSAVE     ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
055676 001414          BEQ      1004$        ;;BRANCH IF OK
055700 042737 000001 177766 BIC      #BIT00,177766     ;;CLEAR THE BIT FOUND SET
055706 113737 001114 056030 MOVB    $ITEMB,IBSAVE     ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
055714 112737 000177 001114 MOVB    #177,$ITEMB        ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
055722 000402          BR      1004$        ;;BRANCH OVER IBSAVE CLEARING

055724 105037 056030          1003$: CLRB IBSAVE          ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
055730          1004$:
055730 032777 020000 123202 BIT      #BIT13,@SWR        ;;SKIP TYPEOUT IF SET
    
```


.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

 *THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH
 *ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (\$ERRTB),
 *AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

056032				\$ERRTYP:		
056032	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
056036	010046			MOV	R0,-(SP)	:::SAVE R0
056040	005000			CLR	R0	:::PICKUP THE ITEM INDEX
056042	153700	001114		BISB	@#\$ITEMB,R0	
056046	001004			BNE	1\$:::IF ITEM NUMBER IS ZERO, JUST
						:::TYPE THE PC OF THE ERROR
056050	013746	001116		MOV	\$ERRPC,-(SP)	:::SAVE \$ERRPC FOR TYPEOUT
						:::ERROR ADDRESS
056054	104402			TYPDC		:::GO TYPE--OCTAL ASCII(ALL DIGITS)
056056	000456			BR	10\$:::GET OUT
056060	122700	000177	1\$:	CMPB	#177,R0	:::SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
056064	001006			BNE	1000\$:::BRANCH IF NOT
056066	113737	001102	056370	MOVB	\$STSTM,PFTSTM	:::GET TEST NUMBER
056074	012700	056230		MOV	#PFECH,R0	:::MOVE POWER FAIL ERROR CALL TABLE TO R0
056100	000406			BR	1001\$:::BRANCH TO CALL ERROR
056102	005300		1000\$:	DEC	R0	:::ADJUST THE INDEX SO THAT IT WILL
056104	006300			ASL	R0	:::
056106	006300			ASL	R0	:::WORK FOR THE ERROR TABLE
056110	006300			ASL	R0	
056112	062700	003630		ADD	#\$ERRTB,R0	:::FORM TABLE POINTER
056116	012037	056126	1001\$:	MOV	(R0)+,2\$:::PICKUP "ERROR MESSAGE" POINTER
056122	001404			BEQ	3\$:::SKIP TYPEOUT IF NO POINTER
056124	104401			TYPE		:::TYPE THE "ERROR MESSAGE"
056126	000000		2\$:	.WORD	0	:::"ERROR MESSAGE" POINTER GOES HERE
056130	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
056134	012037	056144	3\$:	MOV	(R0)+,4\$:::PICKUP "DATA HEADER" POINTER
056140	001404			BEQ	5\$:::SKIP TYPEOUT IF 0
056142	104401			TYPE		:::TYPE THE "DATA HEADER"
056144	000000		4\$:	.WORD	0	:::"DATA HEADER" POINTER GOES HERE
056146	104401	001223		TYPE	,\$CRLF	:::'CARRIAGE RETURN' & 'LINE FEED'
056152	010146		5\$:	MOV	R1,-(SP)	:::SAVE R1
056154	012001			MOV	(R0)+,R1	:::PICKUP "DATA TABLE" POINTER
056156	001415			BEQ	9\$:::BR IF NO DATA TO BE TYPED
056160	012000			MOV	(R0)+,R0	:::PICKUP "DATA FORMAT" POINTER
056162	105720		6\$:	TSTB	(R0)+	:::'OCTAL' OR 'DECIMAL'
056164	001003			BNE	7\$:::BR IF DECIMAL
056166	013146			MOV	@(R1)+,-(SP)	:::SAVE @(R1)+ FOR TYPEOUT
056170	104402			TYPDC		:::GO TYPE--OCTAL ASCII(ALL DIGITS)
056172	000402			BR	8\$	
056174			7\$:			
056174	013146			MOV	@(R1)+,-(SP)	:::SAVE @(R1)+ FOR TYPEOUT
056176	104405			TYPDS		:::GO TYPE--DECIMAL ASCII WITH SIGN
056200	005711		8\$:	TST	(R1)	:::IS THERE ANOTHER NUMBER?
056202	001403			BEQ	9\$:::BR IF NO
056204	104401	056224		TYPE	,11\$:::TYPE TWO(2) SPACES
056210	000764			BR	6\$:::LOOP
056212	012601		9\$:	MOV	(SP)+,R1	:::RESTORE R1
056214	012600		10\$:	MOV	(SP)+,R0	:::RESTORE R0

```
056216 104401 001223          TYPE      ,SCLF          ;:'CARRIAGE RETURN' & 'LINE FEED'  
056222 000207          RTS      PC          ;:RETURN  
056224   040   040   000 11$: .ASCIZ  /  /          ;:TWO(2) SPACES  
                                .EVEN  
056230 056240 056322 056354 PFECH: PFECH1,PFECH2,PFECH3,PFECH4 ;:WORDS DEFINING TABLES BELOW  
056240   120   117   127 PFECH1: .ASCIZ  ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?  
056322   124   105   123 PFECH2: .ASCIZ  ?TESTNO ERR PC CPUERREG?  
                                .EVEN  
056354 056370 001116 056026 PFECH3: .WORD  PFTSTN,$ERRPC,CPSAVE,0  
056364   000   000   000 PFECH4: .BYTE  0,0,0,0  
056370 000000          PFTSTN: .WORD  0          ;:CONTAINS TEST NUMBER FOR PF BIT ERROR
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS    ;;CALL FOR TYPEOUT
*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS

```

```

*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC

```

```

*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPON    ;;CALL FOR TYPEOUT

```

```

*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOC    ;;CALL FOR TYPEOUT

```

```

056372 017646 000000
056376 116637 000001 056615
056404 112637 056617
056410 062716 000002
056414 000406
056416 112737 000001 056615
056424 112737 000006 056617
056432 112737 000005 056614
056440 010346
056442 010446
056444 010546
056446 113704 056617
056452 005404
056454 062704 000006
056460 110437 056616
056464 113704 056615
056470 016605 000012
056474 005003
056476 006105
056500 000404
056502 006105
056504 006105
056506 006105
056510 010503
056512 006103
056514 105337 056616
056520 100016
056522 042703 177770
056526 001002
056530 005704
056532 001403
056534 005204

$TYPOS: MOV @ (SP),-(SP) ;;PICKUP THE MODE
        MOV 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH
        MOV (SP)+, $SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
        ADD #2, (SP) ;;ADJUST RETURN ADDRESS
        BR $TYPON
$TYPOC: MOV #1, $OFILL ;;SET THE ZERO FILL SWITCH
        MOV #6, $SOMODE+1 ;;SET FOR SIX(6) DIGITS
$TYPON: MOV #5, $OCNT ;;SET THE ITERATION COUNT
        MOV R3, -(SP) ;;SAVE R3
        MOV R4, -(SP) ;;SAVE R4
        MOV R5, -(SP) ;;SAVE R5
        MOV $SOMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG R4
        ADD #6, R4 ;;SUBTRACT IT FOR MAX. ALLOWED
        MOV R4, $SOMODE ;;SAVE IT FOR USE
        MOV $OFILL, R4 ;;GET THE ZERO FILL SWITCH
        MOV 12(SP), R5 ;;PICKUP THE INPUT NUMBER
        CLR R3 ;;CLEAR THE OUTPUT WORD
1$: ROL R5 ;;ROTATE MSB INTO 'C'
    BR 3$ ;;GO DO MSB
2$: ROL R5 ;;FORM THIS DIGIT
    ROL R5
    ROL R5
    MOV R5, R3
3$: ROL R3 ;;GET LSB OF THIS DIGIT
    DECB $SOMODE ;;TYPE THIS DIGIT?
    BPL 7$ ;;BR IF NO
    BIC #177770, R3 ;;GET RID OF JUNK
    BNE 4$ ;;TEST FOR 0
    TST R4 ;;SUPPRESS THIS 0?
    BEQ 5$ ;;BR IF YES
4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S

```

056536	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
056542	052703	000040	5\$:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY
056546	110337	056612		MOVB	R3,8\$::SAVE FOR TYPING
056552	104401	056612		TYPE	8\$::GO TYPE THIS DIGIT
056556	105337	056614	7\$:	DECB	\$OCNT	::COUNT BY 1
056562	003347			BGT	2\$::BR IF MORE TO DO
056564	002402			BLT	6\$::BR IF DONE
056566	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
056570	000744			BR	2\$::GO DO THE LAST DIGIT
056572	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
056574	012604			MOV	(SP)+,R4	::RESTORE R4
056576	012603			MOV	(SP)+,R3	::RESTORE R3
056600	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
056606	012616			MOV	(SP)+,(SP)	
056610	000002			RTI		::RETURN
056612	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
056613	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
056614	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
056615	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
056616	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

.SBTTL TRAP DECODER

 *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
 *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 *GO TO THAT ROUTINE.

056620	010046		\$TRAP:	MOV	R0,-(SP)	::SAVE R0
056622	016600	000002		MOV	?(SP),R0	::GET TRAP ADDRESS
056626	005740			TST	-(R0)	::BACKUP BY 2
056630	111000			MOVB	(R0),R0	::GET RIGHT BYTE OF TRAP
056632	006300			ASL	R0	::POSITION FOR INDEXING
056634	016000	056654		MOV	\$TRPAD(R0),R0	::INDEX TO TABLE
056640	000200			RTS	R0	::GO TO ROUTINE

::THIS IS USE TO HANDLE THE 'GETPRI' MACRO

056642	011646		\$TRAP2:	MOV	(SP),-(SP)	::MOVE THE PC DOWN
056644	016666	000004		MOV	4(SP),2(SP)	::MOVE THE PSW DOWN
056652	000002	000002		RTI		::RESTORE THE PSW

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 *BY THE 'TRAP' INSTRUCTION.

		ROUTINE		

056654	056642	\$TRPAD:	.WORD	\$TRAP2
056656	053704		\$TYPE	::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
056660	056416		\$TYPOC	::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
056662	056372		\$TYPOS	::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZERO)
056664	056432		\$TYPON	::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
056666	053460		\$TYPDS	::CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
056670	054606		\$GTSWR	::CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
056672	054516		\$CKSWR	::CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
056674	055060		\$RDCHR	::CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
056676	055150		\$RDLIN	::CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
056700	055330		\$RDOCT	::CALL=RDOCT TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
2 056702	041442		T.SCOP	::CALL=SCOP1 TRAP+13(104413) MY LOCAL SCOPES
3 056704	041514		CHECKT	::CALL=CHECKD TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
4 056706	042032		WAIT.T	::CALL=WAT TRAP+15(104415) WAIT LOOP

.SBTTL ERROR AND MESSAGE TABLE CONDIMITS

2					
3	057072	127	122	117	EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
4	057155	105	122	122	EM2: .ASCIZ /ERROR ON DATA COMMAND/
5	057204	105	122	122	EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/
6	057243	103	117	116	EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/
7	057276	122	105	107	EM14: .ASCIZ /REGISTER FAILED/
8	057316	116	117	116	EM15: .ASCIZ /NON EXISTENT REGISTER, PROGRAM ABORTED./
9	057366	127	101	111	EM16: .ASCIZ /WAIT LOOP FAILED/
10	057407	127	122	111	EM17: .ASCIZ /WRITE CHECK FAILING/
11	057433	122	105	107	EM20: .ASCIZ /REGISTER FAILING/
12	057454	111	116	124	EM21: .ASCIZ /INTERRUPT FAILING/
13	057476	105	122	122	EM22: .ASCII /ERROR ON DRIVES PRESENT -/<CRLF>
14	057530	124	110	105	.ASCII /THE UNIT NO'S FOUND BY SETTING RHAS USING RHER1/<CRLF>
15	057610	050	124	064	.ASCII /((T4) DO NOT AGREE WITH THE UNIT NO'S FOUND/<CRLF>
16	057663	102	131	040	.ASCII /BY LOOKING FOR 'NED' = 0 IN RHCS2 (BIT #12)/<CRLF><LF>
17	057740	116	117	124	.ASCII /NOTE: ON DUAL PORT SYSTEM, A DRIVE ON OTHER PORT WILL /<CRLF>
18	060027	116	117	124	.ASCII /NOT GIVE 'NED', BUT WILL GIVE RHAS RESPONSES/<CRLF>
19	060104	110	105	116	.ASCIZ /HENCE THERE WILL BE AN EXTRA DRIVE/
20	060147	114	117	117	EM24: .ASCIZ /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
21	060242	114	117	117	EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/
22	060302	103	125	122	EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<CRLF>
23	060374	101	106	124	.ASCIZ /AFTER A SEEK AND INIT/
24	060422	105	103	103	EM31: .ASCII /ECC GENERATED IS INCORRECT/<CRLF>
25	060455	105	126	105	.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN 'DATA USED'/
26	060544	117	116	040	EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<CRLF>
27	060630	105	103	103	.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<CRLF>
28	060674	117	116	114	.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<CRLF>
29	060753	124	110	111	.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
30	061030	110	111	107	EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
31	061102	132	105	122	EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
32	061175	120	117	123	EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<CRLF>
33	061270	114	117	127	.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<CRLF>
34	061355	061	061	040	.ASCII /11 BITS OF GOOD ECC1/<CRLF>
35	061402	104	101	124	.ASCIZ /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
36	061471	117	116	040	EM36: .ASCIZ /ON READ COMMAND WITH NON-CORRECTABLE ERROR 'DCK' AND 'ECH SHOULD BE SET/
37	061602	120	122	117	EM37: .ASCII /PROGRAM ERROR BIT #10 IN RHCS2 DID NOT SET/<CRLF>
38	061655	111	106	040	.ASCIZ /IF POSITION REGISTER =10040 OR 10041, IT IS GOOD/
39					
40	061736	122	110	127	EM40: .ASCII /RHWC DID NOT = 0 UPON COMPLETION OF READ/<CRLF>
41	062007	117	122	040	.ASCIZ /OR WRITE HEADER AND DATA/

ERROR AND MESSAGE TABLE CONDIMENTS

1	062040	200	101	114	PREAMB: .ASCII	<CRLF>/ALL DCL'S UNDER TEST MUST BE LOCKED ON CORRECT PORT./
2	062125	200	111	106	.ASCII	<CRLF>/IF CHANGES ARE REQUIRED ON PORT SWITCH, THEN A CYCLE/
3	062212	200	125	120	.ASCII	<CRLF>/UP SEQUENCE IS REQUIRED FOR STROBING THE PORT SELECT/
4	062277	200	106	114	.ASCII	<CRLF>/FLOP./
5	062305	200			.ASCII	<CRLF>
6	062306	200	101	114	.ASCII	<CRLF>/ALL DCL'S NOT UNDER TEST MUST BE SWITCHED OFF OR/
7	062373	200	114	117	.ASCII	<CRLF>/LOCKED ON THE OTHER PORT./
8						
9	062426	200	116	117	NDRVAS: .ASCII	<CRLF>/NO DRIVES PRESENT, RHAS=0/
10	062460	200			.ASCII	<CRLF>
11	062461	200	127	122	.ASCII	<CRLF>/WRITTING ONES INTO RHER1 FOR ALL UNIT NUMBERS DOES/
12	062546	200	116	117	.ASCII	<CRLF>/NOT SET ANY BIT IN RHAS, SO ABORT PROGRAM./
13	062621	200			.ASCII	<CRLF>
14	062622	200	124	117	.ASCII	<CRLF>/TO LOOP ON THIS TEST WITHOUT PRINTOUT, SET SWITCHES/
15	062707	200	061	063	.ASCII	<CRLF>/13, 8 AND 2./<CRLF>
16						
17	062726	106	101	124	CPHALT: .ASCII	/FATAL ERROR - SEE DOCUMENT LISTING/<CRLF>
18	062771	040	200	207	.ASCII	/ /<CRLF><207><377><377><207><377><377><207><377><377>
19	063004	124	110	105	.ASCII	/THE CONTROLLER OR DEVICE HAS GONE OFFLINE, LOST/<CRLF>
20	063064	047	122	105	.ASCII	/'READY', BECOME UNAVAILABLE, OR HAS STATUS BITS/<CRLF>
21	063144	127	110	111	.ASCII	/WHICH CANNOT BE CLEARED/
22						
23	063174	040			BLNKS8: .ASCII	/ /
24	063175	040			BLNKS7: .ASCII	/ /
25	063176	040			BLNKS6: .ASCII	/ /
26	063177	040			BLNKS5: .ASCII	/ /
27	063200	040			BLNKS4: .ASCII	/ /
28	063201	040			BLNKS3: .ASCII	/ /
29	063202	040			BLNKS2: .ASCII	/ /
30	063203	040	000		BLNKS1: .ASCII	/ /

1	067540	001116	001432	041130	DT1:	.WORD	SERRPC,TSTNM,REGADR,\$GDDAT,\$BDDAT,0
2	067554	001116	001432	046426	DT2:	.WORD	SERRPC,TSTNM,ERWORD,\$GDDAT,0
3	067566	001116	001432	046426	DT3:	.WORD	SERRPC,TSTNM,ERWORD,\$GDDAT,\$BDDAT,0
4							
5	067602	001116	001432	001122	DT11:	.WORD	SERRPC,TSTNM,\$BDADR,CS1,CS2,DS1,ER1,0
6	067622	001116	001432	001122	DT14:	.WORD	SERRPC,TSTNM,\$BDADR,\$BDDAT,CS1,CS2,DS1,ER1,0
7	067644	001116	001432	001200	DT15:	.WORD	SERRPC,TSTNM,\$TMP1,0
8	067654	001116	001432	001204	DT16:	.WORD	SERRPC,TSTNM,\$TMP3,\$TMP1,\$TMP0,\$BDDAT,0
9	067672	001116	001432	001310	DT17:	.WORD	SERRPC,TSTNM,BA,DB,WC,CS1,CS2,0
10							
11	067712	001116	001432	001316	DT20:	.WORD	SERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0
12	067732	001116	001432	001314	DT21:	.WORD	SERRPC,TSTNM,CS1,AS,DS1,0
13	067746	001116	001432	000000	DT22:	.WORD	SERRPC,TSTNM,0
14	067754	001116	001432	001320	DT24:	.WORD	SERRPC,TSTNM,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3,0
15	067774	001116	001432	001414	DT26:	.WORD	SERRPC,TSTNM,PCJSR,\$BDADR,CS1,CS2,DS1,ER1,0
16	070016	001116	001432	001414	DT27:	.WORD	SERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0
17							
18	070034	001116	001432	001414	DT30:	.WORD	SERRPC,TSTNM,PCJSR,REGADR,\$GDDAT,\$BDDAT,0
19	070052	001116	001432	046426	DT31:	.WORD	SERRPC,TSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0
20	070074	001116	001432	001414	DT32:	.WORD	SERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1,0
21	070120	001116	001432	001414	DT33:	.WORD	SERRPC,TSTNM,PCJSR,ERWORD,\$GDDAT,CS1,DS1,ER1,0
22	070142	001116	001432	044154	DT34:	.WORD	SERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0
23	070162	001116	001432	044154	DT35:	.WORD	SERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSITI,ER1,0
24	070204	001116	001432	001414	DT36:	.WORD	SERRPC,TSTNM,PCJSR,MR,EC1,EC2,0
25	070222	001116	001432	001344	DT37:	.WORD	SERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE,0
26							
27	070246	001116	001432	001126	DT40:	.WORD	SERRPC,TSTNM,\$BDDAT,0
28							
29	070256	000	000	000	DF1:	.BYTE	0,0,0,0,0
30	070263	000	000	001	DF2:	.BYTE	0,0,1,0
31	070267	000	000	001	DF3:	.BYTE	0,0,1,0,0
32							
33	070274	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0,0
34	070303	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0,0
35	070313	000	000	000	DF15:	.BYTE	0,0,0
36	070316	000	000	000	DF16:	.BYTE	0,0,0,0,0
37	070323	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0,0
38							
39	070332	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0,0
40	070341	000	000	000	DF21:	.BYTE	0,0,0,0,0
41	070346	000	000	000	DF22:	.BYTE	0,0,0,0
42	070352	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0,0
43	070361	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0,0
44	070371	000	000	000	DF27:	.BYTE	0,0,0,0,0,0
45							
46	070377	000	000	000	DF30:	.BYTE	0,0,0,0,0,0,0
47	070405	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0,0
48	070415	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0,0
49	070426	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0,0
50	070436	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0,0
51	070445	000	000	000	DF35:	.BYTE	0,0,0,0,0,0,0,0,0
52	070455	000	000	000	DF36:	.BYTE	0,0,0,0,0,0,0
53	070463	000	000	000	DF37:	.BYTE	0,0,0,0,0,0,0,0,0,0
54							
55	070474	000	000	000	DF40:	.BYTE	0,0,0
56						.EVEN	
57							

58 000200 .END 200

SYMBOL TABLE

ABS = 000200	BLNKS6 063176	DF31 070405	DT22 067746	ERWORD 046426
ACL = 000040	BLNKS7 063175	DF32 070415	DT24 067754	ER1 001316
ACU = 100000	BLNKS8 063174	DF33 070426	DT26 067774	ER2 001322
ADTIMO 046034	BLT1 041160	DF34 070436	DT27 070016	ER3 001330
AOE = 001000	BLT2 041202	DF35 070445	DT3 067566	EXT1 = 000001
AS 001332	BLT3 041212	DF36 070455	DT30 070034	EXT10 = 000010
ATA = 100000	BPTVEC= 000014	DF37 070463	DT31 070052	EXT2 = 000002
ATABLE 003620	CA 001326	DF40 070474	DT32 070074	EXT20 = 000020
ATTENT 001416	CAT 035652	DF5 = 000001	DT33 070120	EXT4 = 000004
ATO = 000001	CC 001352	DH1 063205	DT34 070142	EXT40 = 000040
AT1 = 000002	CHECKD= 104414	DH11 063451	DT35 070162	FEN = 000200
AT2 = 000004	CHECKE 041704	DH14 063630	DT36 070204	FER = 000020
AT3 = 000010	CHECKT 041514	DH15 064027	DT37 070222	FILLEC 044750
AT4 = 000020	CKSWR = 104407	DH16 064060	DT40 070246	FMT22 = 010000
AT5 = 000040	CLAREA 041376	DH17 064222	DVA = 004000	FNWORD 051504
AT6 = 000100	CLDISK 041460	DH2 063327	ECDATA 044152	FORMAT 047670
AT7 = 000200	CLR = 000040	DH20 064404	ECH = 000100	FOUT 051510
A16 = 000400	COMHD 046146	DH21 064566	ECI = 004000	FRMAT1 032632
A17 = 001000	COMPA 047220	DH22 064705	ECORR 044576	FSYNER 051460
BA 001310	COMPAR 042354	DH24 065007	ECTEST 044210	FUTABL 001442
BADTMO 004230	COMWHD 051270	DH26 065171	EC1 001344	GCRC 051454
BAI = 000010	COUNTD 047666	DH27 065373	EC2 001346	GECC1 044154
BASECH 045104	CPHALT 062726	DH30 065535	EMTVEC= 000030	GECC2 044156
BEGIN 004310	CPSAVE 056026	DH31 065677	EM1 057072	GO = 000001
BEGIN2 004316	CR = 000015	DH32 066101	EM11 057243	GRV = 000010
BITST 041132	CRC 042666	DH33 066323	EM14 057276	GTSWR = 104406
BIT0 = 000001	CRLF = 000200	DH34 066525	EM15 057316	HADTMP 044176
BIT00 = 000001	CSF = 000002	DH35 066707	EM16 057366	HARDER 044170
BIT01 = 000002	CSU = 000010	DH36 067111	EM17 057407	HCCRCE 043270
BIT02 = 000004	CS1 001314	DH37 067247	EM2 057155	HCE = 000200
BIT03 = 000010	CS2 001312	DH40 067464	EM20 057433	HCI = 002000
BIT04 = 000020	CYL 046306	DIGB = 000004	EM21 057454	HCRC = 000400
BIT05 = 000040	DAREAD 046362	DISK 050224	EM22 057476	HDESYN 051470
BIT06 = 000100	DATENV 044172	DISPLA 001142	EM24 060147	HDWSYN 050222
BIT07 = 000200	DAWORD 046366	DISPRE 000174	EM25 060242	HEADER 050176
BIT08 = 000400	DB 001304	DLT = 100000	EM30 060302	HEDGAP 046322
BIT09 = 001000	DCK = 100000	DL64 = 000020	EM31 060422	HEDSYN 046324
BIT1 = 000002	DCLEAR 001450	DMD = 000001	EM32 060544	HEGAP 050210
BIT10 = 002000	DDISP = 177570	DOG 035206	EM33 061030	HT = 0C 7011
BIT11 = 004000	DENVL = 000200	DPR = 000400	EM34 061102	IAE = 002000
BIT12 = 010000	DE1 = 000040	DRY = 000200	EM35 061175	IBSAVE 056030
BIT13 = 020000	DF1 = 070256	DST = 001320	EM36 061471	IE = 000100
BIT14 = 040000	DF11 070274	DSWR = 177570	EM37 061602	ILF = 000001
BIT15 = 100000	DF14 070303	DS1 001336	EM40 061736	ILLEGL 001504
BIT2 = 000004	DF15 070313	DT 001340	EM6 057204	ILR = 000002
BIT3 = 000010	DF16 070316	DTAGAP 051230	ERCLFC 017436	IOTVEC= 000020
BIT4 = 000020	DF17 070323	DTE = 010000	ERCRC 051464	IR = 000100
BIT5 = 000040	DF2 070263	DTSY = 001000	ERCSC2C 012302	IXE = 004000
BIT6 = 000100	DF17 070323	DT1 067540	ERFLG\$ 001406	JP1 036164
BIT7 = 000200	DF20 070332	DT11 067602	ERHDGP 051466	JP2 036200
BIT8 = 000400	DF21 070341	DT14 067622	ERHEAD 051462	KEY1 046312
BIT9 = 001000	DF22 070346	DT15 067644	ERPOS 044574	KEY2 046314
BLNKS1 063203	DF24 070352	DT16 067654	ERR = 040000	KIPAR0= 172340
BLNKS2 063202	DF26 070361	DT17 067672	ERROR = 104000	KIPAR1= 172342
BLNKS3 063201	DF27 070371	DT2 067554	ERRVEC= 000004	KIPAR2= 172344
BLNKS4 063200	DF3 070267	DT20 067712	ERSTAR 046120	KIPAR3= 172346
BLNKS5 063177	DF30 070377	DT21 067732	ERUNIT 046116	KIPAR4= 172350

SYMBOL TABLE

KIPAR5=	172352	OR	=	000200	PRER1	010166	RHEC2	001272	SR2	=	177576			
KIPAR6=	172354	OUT	=	046372	PRER2	010464	RHER1	001242	SR3	=	172516			
KIPAR7=	172356	PAR	=	000010	PRER3	010640	RHER2	001246	SSECTR	=	051530			
KIPDR0=	172300	PAT	=	000020	PROF	010530	RHER3	001254	SSYN	=	046320			
KIPDR1=	172302	PCJSR	=	001414	PROG	=	001000	RHLA	001274	STACK	=	001100		
KIPDR2=	172304	PFECH	=	056230	PRWC	=	010056	RHMR	001260	START	=	004324		
KIPDR3=	172306	PFECH1	=	056240	PRO	=	000000	RHOF	001250	STKLMT=	177774			
KIPDR4=	172310	PFECH2	=	056322	PR1	=	000040	RHSN	001266	SWR	=	001140		
KIPDR5=	172312	PFECH3	=	056354	PR2	=	000100	RHWC	001232	SWREG	=	000176		
KIPDR6=	172314	PFECH4	=	056364	PR3	=	000140	RH70	001436	SW0	=	000001		
KIPDR7=	172316	PFTSTN	=	056370	PR4	=	000200	RH70CK	004736	SW00	=	000001		
LA	001350	PGE	=	002000	PR5	=	000240	RKEY1	046416	SW01	=	000002		
LAD	041440	PIE1	=	100000	PR6	=	000300	RKEY2	046420	SW02	=	000004		
LERR	041126	PIE10	=	000100	PR7	=	000340	RMR	=	000004	SW03	=	000010	
LF	=	PIE11	=	000040	PS	=	177776	RNCTR1	051266	SW04	=	000020		
LST	=	PIE12	=	000020	PSEL	=	002000	RNO	052624	SW05	=	000040		
MAKECY	044032	PIE13	=	000010	PSU	=	000001	RNWAT1	051346	SW06	=	000100		
MASK	041124	PIE14	=	000004	PSW	=	177776	RPVEC	001226	SW07	=	000200		
MCLK	=	PIE15	=	000002	PUTREG	041064	RPVECT	052740	SW08	=	000400			
MCPE	=	PIE16	=	000001	PWRVEC=	000024	RP06	001434	SW09	=	001000			
MHS	=	PIE17	=	100000	P12	044202	RSETR	046414	SW1	=	000002			
MID	044022	PIE18	=	040000	P22	044204	RSYNC	046410	SW10	=	002000			
MIDDLE	043720	PIE19	=	020000	P24	044206	RUNCTR	046144	SW11	=	004000			
MINX	=	PIE2	=	040000	P3	044200	RUNWAT	046224	SW12	=	010000			
MMVEC	=	PIE20	=	010000	RAW	=	000020	R6	=	%000006	SW13	=	020000	
MOL	=	PIE21	=	004000	RCOM	052626	R7	=	%000007	SW14	=	040000		
MPE	=	PIE22	=	002000	RCYL	046412	SAVDT	001410	SW15	=	100000			
MR	001334	PIE23	=	001000	RDCHR	=	104410	SAVER	042152	SW2	=	000004		
MRD	=	PIE24	=	000400	RDHEAD	046430	SAVSN	001412	SW3	=	000010			
MSE	=	PIE25	=	000200	RDLIN	=	104411	SC	=	100000	SW4	=	000020	
MSTCK	=	PIE26	=	000100	RDOCT	=	104412	SCOPE	=	000004	SW5	=	000040	
MWR	=	PIE27	=	000040	RDY	=	000200	SCOPI	=	104413	SW6	=	000100	
MXF	=	PIE28	=	000020	READ	047222	SCRC	051536	SW7	=	000200			
NCODE	044162	PIE29	=	000010	READAT	001466	SCYL	051526	SW8	=	000400			
NCOUNT	044164	PIE3	=	020000	READIN	001502	SC1	=	000100	SW9	=	001000		
NDRVAS	062426	PIE30	=	000004	RECALI	001446	SC10	=	001000	TAGDTE	=	001430		
NED	=	PIE31	=	000002	REDATA	052630	SC2	=	000200	TBITVE=	000014			
NEM	=	PIE32	=	000001	REFOR	001470	SC20	=	002000	TDF	=	000040		
NHS	=	PIE4	=	010000	REGADR	041130	SC4	=	000400	TESDTE	=	001426		
NOPERA	001442	PIE5	=	004000	REGSA1	055504	SEARCH	052354	TESTAD	=	040306			
NOSYNC	046422	PIE6	=	002000	REINTO	002554	SECGAP	050126	TIMCNT	=	042030			
NOUNIT	001376	PIE7	=	001000	RELEAS	001452	SECOTR	046310	TKVEC	=	000060			
NOWORD	046354	PIE8	=	000400	RESVEC=	000010	SECTR	052352	TMPILL	=	001422			
NUNIT	001400	PIE9	=	000200	RETCL	001476	SEECOM	001472	TOLGAP	=	051232			
OCYL	=	PIP	=	020000	RHAS	001256	SEGPER	051456	TOTALA	=	001420			
OF	001324	PIRQ	=	177772	RHBA	001234	SELECT	001402	TPVEC	=	000064			
OFREV	=	PIRQVE=	000240	PKACK	001500	RHBAE	001300	SELTST	006424	TRAPVE=	000034			
OFSETC	001474	PLU	=	020000	RHCA	001252	SERCH	001454	TRK	=	040000			
OF100	=	POSITI	044166	RHCC	001276	RHCS1	001240	SETDSK	043142	TRK1	=	004000		
OF200	=	PRBA	010122	RHCS2	001236	RHCS2	001236	SILOSZ	001440	TRK10	=	040000		
OF25	=	PRCA	010574	RHCS3	001302	RHCS3	001302	SILOTB	050224	TRK2	=	010000		
OF400	=	PRCS1	007642	RHDB	001230	RHDB	001230	SKEY1	051532	TRK20	=	100000		
OF50	=	PRCS2	007576	RHDST	001244	RHDST	001244	SKEY2	051534	TRK4	=	020000		
OF800	=	PRDST	010420	RHDS1	001262	RHDS1	001262	SKI	=	040000	TRTVEC=	000014		
OPDCL	=	PRE	=	000020	RHDT	001264	RHDT	001264	SN	=	001342	TSECC	=	001424
OPERSE	040310	PREAMB	062040	RHEC1	001270	RHEC1	001270	SRO	=	177572	TSECCG	=	044160	
OP!	=							SR1	=	177574				

SYMBOL TABLE

TSTNM	001432	TST45	017464	VV	= 000100	\$DBLK	053674	\$REG1	001164
TST1	005052	TST46	020402	WAIT.T	042032	\$DOAGN	040262	\$REG2	001166
TST10	007620	TST47	021056	WAT	= 104415	\$DTBL	053664	\$REG3	001170
TST100	033444	TST5	006440	WC	= 001306	\$ENDAD	040252	\$REG4	001172
TST101	033500	TST50	021464	WCE	= 040000	\$ENDCT	040220	\$REG5	001174
TST102	033622	TST51	022030	WCF	= 000040	\$ENDMG	040271	\$RTNAD	040264
TST103	033656	TST52	022374	WCRC	= 050206	\$ENULL	040266	\$SAVR6	057060
TST104	034000	TST53	022552	WCU	= 000001	\$EOP	040164	\$SCOPE	053020
TST105	034034	TST54	022606	WCYL	= 051444	\$EOPCT	040212	\$SETUP=	000117
TST106	034134	TST55	023474	WECC1	= 051224	\$ERFLG	001103	\$SS1	= 000000
TST107	034170	TST56	023530	WECC2	= 051226	\$ERMAX	001115	\$STUP	= 177777
TST11	007664	TST57	024040	WKEY1	= 051450	\$ERROR	055470	\$SVLAD	053414
TST110	034270	TST6	007202	WKEY2	= 051452	\$ERRPC	001116	\$SVPC	= 000214
TST111	034416	TST60	024376	WLE	= 004000	\$ERRTB	003630	\$SWR	= 167770
TST112	034536	TST61	024736	WORD	= 047216	\$ERRTY	056032	\$SWRMK=	000000
TST113	034572	TST62	025202	WRCHDA	= 042516	\$ERTTL	001112	\$TIMES	001212
TST114	035206	TST63	025566	WRCHIT	= 001460	\$ESCAP	001214	\$TKB	001146
TST115	035242	TST64	026152	WRCHK	= 001456	\$FILLC	001156	\$TKCNT	054176
TST116	035652	TST65	026536	WRCHHD	= 042204	\$FILLS	001155	\$TKINT	054216
TST117	036476	TST66	027076	WRDATA	= 047674	\$GDADR	001120	\$TKQEN=	054215
TST12	010034	TST67	027442	WRFROM	= 001510	\$GDDAT	001124	\$TKQIN	054200
TST120	036532	TST7	007530	WRHEAD	= 051540	\$GET42	040242	\$TKQOU	054202
TST121	037064	TST70	027772	WRIDAT	= 001462	\$GTSWR	054606	\$TKQSR	054204
TST122	037410	TST71	030300	WRIFOR	= 001464	\$HD	= 000000	\$TKS	001144
TST123	037740	TST72	030670	WRITE	= 047454	\$HIOCT	055466	\$TKSRV	054266
TST13	01 100	TST73	031234	WRL	= 004000	\$ICNT	001104	\$TMP0	001176
TST14	010 44	TST74	031600	WRU	= 000400	\$ILLUP	057054	\$TMP1	001200
TST15	010 10	TST75	032206	WSECTR	= 051446	\$INTAG	001135	\$TMP2	001202
TST16	010376	TST76	032610	WSSYNC	= 050174	\$ITEMB	001114	\$TMP3	001204
TST17	010442	TST77	033140	WSU	= 000004	\$LF	001224	\$TMP4	001206
TST2	005534	TUF	= 000100	WTRK	= 051402	\$LPADR	001106	\$TMP5	001210
TST20	010506	TY	= 046424	WWORD	= 047452	\$LPERR	001110	\$TN	= 000124
TST21	010552	TYPDS	= 104405	X	= 046316	\$MNEW	055317	\$TPB	001152
TST22	010616	TYPE	= 104401	XE2	= 006076	\$MSWR	055306	\$TPFLG	001157
TST23	010662	TYPOC	= 104402	Y	= 046356	\$MXCNT	053456	\$TPS	001150
TST24	011270	TYPON	= 104404	ZCODE	= 044174	\$NULL	001154	\$TRAP	056620
TST25	012314	TYPOS	= 104403	ZER	= 000400	\$NWTST=	000001	\$TRAP2	056642
TST26	012456	T.SCOP	= 041442	ZWORDS	= 047672	\$OCNT	056614	\$TRP	= 000016
TST27	C:3026	UN	= 005610	\$AUTOB	= 001134	\$OMODE	056616	\$TRPAD	056654
TST3	005620	UNIT	= 001374	\$BDADR	= 001122	\$OVER	053442	\$TSTNM	001102
TST30	013204	UNITS	= 001354	\$BDDAT	= 001126	\$PASS	001100	\$TTYIN	055256
TST31	013432	UNITSL	= 001404	\$BELL	= 001216	\$POWER	057062	\$TYPDS	053460
TST32	013672	UNLOAD	= 001444	\$CHARC	= 054172	\$PWRAD	057050	\$TYPE	053704
TST33	014016	UNS	= 040000	\$CKSWR	= 054516	\$PWRDN	056710	\$TYPEC	054054
TST34	014316	UPE	= 020000	\$CMTAG	= 001100	\$PWRMG	057044	\$TYPEX	054174
TST35	014454	US1	= 000001	\$CM1	= 000006	\$PWRUP	056762	\$TYPOC	056416
TST36	014574	US2	= 000002	\$CM2	= 000014	\$QUES	001222	\$TYPON	056432
TST37	014752	US4	= 000004	\$CM3	= 000006	\$RDCHR	055060	\$TYPOS	056372
TST4	005672	UWR	= 000010	\$CM4	= 000006	\$RDLIN	055150	\$XOFF	= 000023
TST40	015076	VAR1	= 036146	\$CNTLC	= 055267	\$RDOCT	055330	\$XON	= 000021
TST41	015222	VAR2	= 036154	\$CNTLG	= 055301	\$RDSZ	= 000011	\$XTSTR	053056
TST42	015516	VUF	= 000002	\$CNTLU	= 055274	\$RECAD	001160	\$SGET4=	000000
TST43	015742	VU30	= 010000	\$CRLF	= 001223	\$REG0	001162	\$OFILL	056615
TST44	016332								

. ABS. 070500 000
000000 001

CZRJGE RPO4/5/6 DSKLS PT1 MACRO V04.00 17-NOV-81 16:32:34 PAGE 54-5^N 2
SYMBOL TABLE

SEQ 0232

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 57856 WORDS (226 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 71 PAGES
CZRJGE.BIC,CZRJGE=CZRJGE.DOC,CZRJGE,[20,0]SYSMAC/M