

RH70/RP04

MECHANICAL R/W TEST
MD-11-DERPK-C

EP-DERPK-C-DL-A

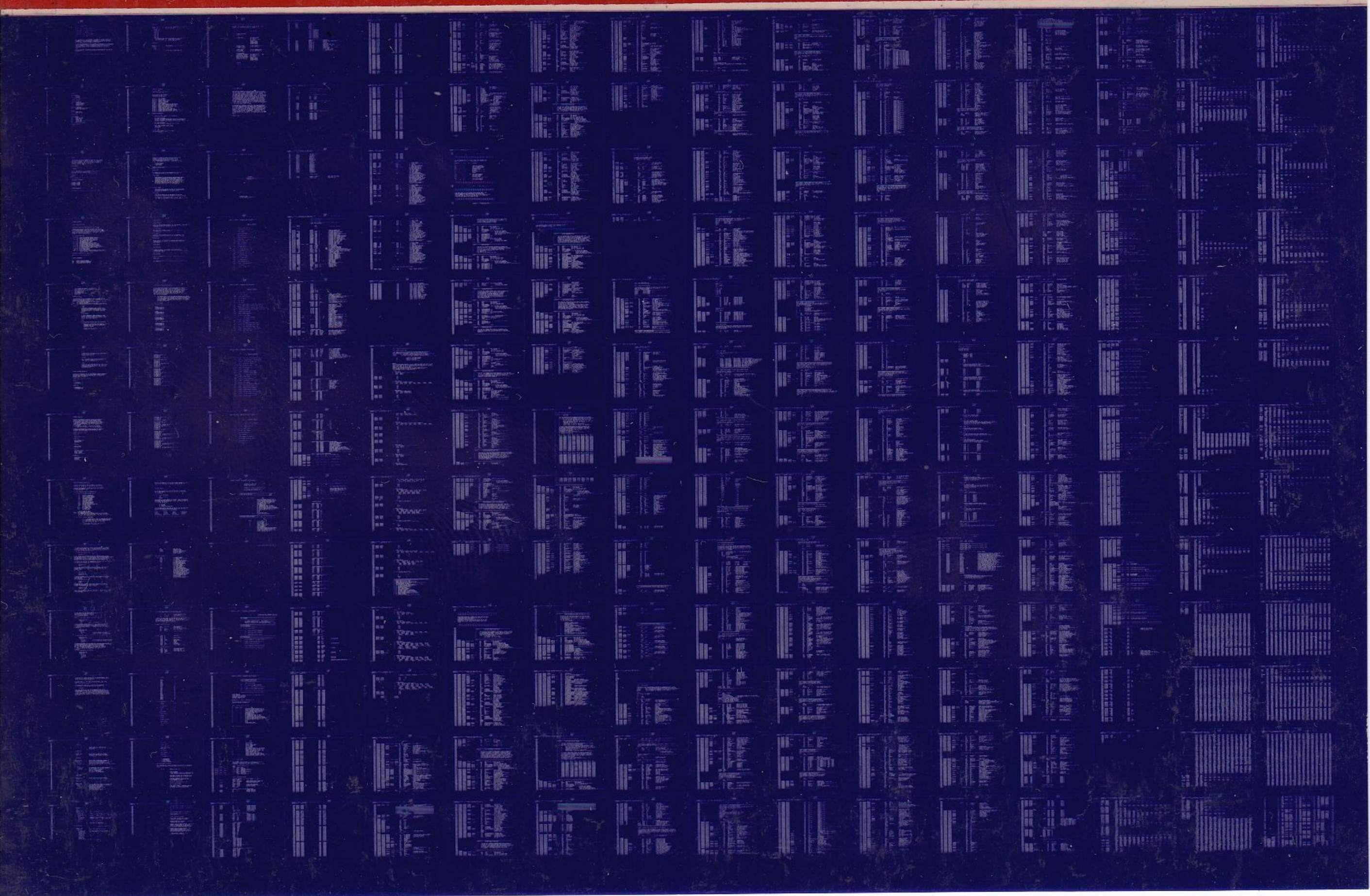
NOV 1976

COPYRIGHT © 1976

digital

FICHE 1 OF 2

MADE IN USA



RH70/RP04

MECHANICAL R/W TEST
MD-11-DERPK-C

EP-DERPK-C-DL-A
COPYRIGHT © 1976
FICHE 2 OF 2

NOV 1976
digital
MADE IN USA

B01

MAINDEC-11-DEPK-C "MECHANICAL AND READ/WRITE TEST"
DEPK-C.P11

MAY11 27(732) 14-OCT-76 10:36 PAGE 2

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DEPK-C-D

PRODUCT NAME: RH70/RP04 MECHANICAL AND READ/WRITE TEST

DATE CREATED: OCTOBER 21, 1975

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: J. LACEY/C. HESS/SUB MALLICK

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (C) 1974, 1975 BY DIGITAL EQUIPMENT CORPORATION

MAINDEC-11-DEPK-C "MECHANICAL AND READ/WRITE TEST"
DEPK-C.P11

CONTENTS

- 1. ABSTRACT
- 2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
- 3. LOADING PROCEDURE
- 4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 OPERATOR ACTION
 - 4.4 PROGRAM ACTION
- 5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACTS
- 6. ERRORS
 - 6.1 ERROR TYPES
 - 6.2 ERROR RECOVERY
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 COMMON TAGS
 - 8.4 TIMING TESTS (TST7-TST12) PRINTOUTS
 - 8.5 END OF TEST
 - 8.6 RH70/RP04 DRIVER
- 9. PROGRAM DESCRIPTION
- 10. LISTING

72
71
70
69
68
67
66
65
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1

110
109
108
107
106
105
104
103
102
101
100
099
098
097
096
095
094
093
092
091
090
089
088
087
086
085
084
083
082
081
080
079
078
077
076
075
074
073

1. ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT WILL INSURE THE DISK IS CAPABLE OF PERFORMING SEEKS, THE ACCESS TIMES ARE WITHIN TOLERANCE, THE TRACK AND SECTOR ADDRESSING CIRCUITRY OPERATES PROPERLY, AND THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONAL.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH 16K OF MEMORY, CONSOLE TELETYPE, RH70/RP04 DISK WITH FORMATTED PACK, AND A KW11-P CLOCK (IF TIMING TESTS ARE DESIRED).

2.2 STORAGE

THIS PROGRAM WILL LOAD IN 12K BUT REQUIRES. 16K TO RUN

2.3 PRELIMINARY PROGRAMS

MAINDEC-11-DERPS
MAINDEC-11-DERPT
MAINDEC-11-DERPU
MAINDEC-11-DERPV

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
1574. STARTING PROCEDURE4.1 CONTROL SWITCH SETTINGS

THE CONTROL SWITCH SETTINGS ARE INPUT VIA THE TTY.

TO ENTER THE CONTROL SWITCH SETTING MODE PLACE SW<07>=1 BEFORE PRESSING START. THEN UPON STARTING THE PROGRAM IT WILL TYPE THE PRESENT CONTENTS OF THE CONTROL SWITCH REGISTER (C.SWR) AND WAIT FOR THE NEW SETTING TO BE INPUT. THE INPUT STRING MUST CONSIST OF A SLASH (/), 1 TO 6 OCTAL DIGITS, TWO PERIODS (...) AND A CARRIAGE RETURN.

THE C.SWR SETTINGS ARE:

C.SWR<15>=0...WRITE PACK BEFORE TESTING (TEST16)
 =1...INHIBIT WRITE PACK BEFORE TESTING (TEST16)
C.SWR<14>=0...NO STALL BETWEEN DRIVE FUNCTIONS
 =1...STALL AFTER EVERY DRIVE FUNCTION
C.SWR<13>=0...USE SPECIFIC STALL TIMES
 =1...USE RANDOM STALL TIMES
C.SWR<12>=0...NO INCREMENTING STALLS IN TEST4
 =1...PERFORM INCREMENTING STALLS IN TEST4
C.SWR<08>=0...DO IMPLIED SEEKS WITH DATA TRANSFERS
 =1...DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
C.SWR<07>=0...DO READ HEADER AND DATA COMMANDS IN TESTS 0-6
 =1...DO EXPLICIT SEEK COMMANDS IN TESTS 0-6
C.SWR<06>=0...60 HZ POWER SOURCE
 =1...50 HZ POWER SOURCE
C.SWR<05>=0...ALLOW SOFTWARE TIMEOUTS(ENABLE WATCHDOG TIMER)
 =1...INHIBIT SOFTWARE TIMEOUTS(DISABLE WATCHDOG TIMER)

THE DEFAULT CONDITION OF C.SWR<15:00>=0.

REFER TO 4.4.1 FOR C.SWR SELECTION

4.2 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS
204 SELECT OPERATING PARAMETERS
210 SELECT RH70/RP04 ADDRESSES
214 COMBINATION OF 204 AND 210

158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204

-
1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3.)
 2. LOAD A FORMATTED PACK INTO DRIVE(S) TO BE TESTED
 3. BRING DRIVE(S) TO ONLINE STATE, WRITE ENABLED, AND LOCKED ON PORT.
 4. LOAD ADDRESS 200.
 5. SET SWITCHES (SEE SECTION 4.1 & 5.1)
 6. PRESS START.

4.4 PROGRAM ACTION

IN AN EFFORT TO ALLOW CONVERSATION WITH A PROGRAM FOR THE PURPOSE OF CONTROLLING ITS OPERATION AND PARAMETERS THE FOLLOWING CONSTRUCTIONS HAVE BEEN ADOPTED.

NOTE1: IN ALL EXAMPLES BRACKETS ARE USED FOR CLARITY AND ARE NOT TYPED BY THE USER.

NOTE2: THE CARRIAGE RETURN TYPED BY THE USER IS INDICATED BY <CR> AND WILL BE ECHOED AS A "CARRIAGE RETURN-LINE FEED."

<.><CR> PERIOD

A STATEMENT TERMINATOR: WHEN TYPED AT THE END OF A LINE (LEGAL ON ALL LINES) IT TELLS THE PARAMETER STRING INTERPRETER (PSI) THIS IS THE END OF CHANGES TO THE CURRENT PARAMETER STRING.

<..><CR> PERIOD PERIOD

AN INPUT TERMINATOR: WHEN A PERIOD IS TYPED AFTER A "STATEMENT TERMINATOR PERIOD" IT TELLS THE PSI THIS IS THE END OF ALL CHANGES (LEGAL ON ALL LINES) BEGIN EXECUTION.

<,><CR> COMMA

A CONTINUATION INDICATOR AND FIELD SEPARATOR:

- 1) THIS IS THE CONSTRUCTION USED WHEN NOT USING A PERIOD: THAT IS TO SAY THE COMMA SAYS "TERMINATE THE LINE BUT NOT THE INPUT FOR PARAMETER CHANGES".
- 2) SEPARATE MULTIPLE ENTRIES ON THE SAME TEXT STRING, I.E., A,B,C,D.

205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245

</> SLASH
 A MODIFICATION INDICATOR: AT ANY GIVEN PARAMETER STOP IF A SLASH IS TYPED IT SAYS "OPEN THIS PARAMETER FOR CHANGES".

<↑U> CONTROL-U
 DUMP THE PRESENT INPUT STRING AND START A NEW LINE. TYPED BY DEPRESSING THE "CONTROL KEY" (CTRL) AND THEN STRIKING THE "U".

<\> RUBOUT
 DELETE THE LAST CHARACTER FROM THE INPUT STRING. TYPED BY STRIKING THE "RUBOUT" KEY. WHICH WILL BE ECHOED BY A BACKSLASH (\) FOLLOWED BY THE CHARACTER DELETED.

4.4.1 CONTROL SWITCHES SELECTION

STARTING THE PROGRAM AT ANY OF THE POSSIBLE STARTING ADDRESSES WITH SW<07>=1 WILL RESULT IN ENTERING THE "CONTROL SWITCH SETTING" MODE. THUS, ALLOWING THE OPERATOR TO SPECIFY THE DESIRED STATE OF "C.SWR".

4.4.1.1 CONTROL SWITCH SELECTION EXAMPLES

EXAMPLE #1

SET SW<07>=0
C.SWR=000000/400..

EXAMPLE #2

SET SW<07>=0
C.SWR=000000/220.
C.SWR=000000/220..

4.4.2 ADDRESS SELECTION

STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC SELECT OF THE DEFAULT VALUES OF BUS ADDRESS (RHCS1), VECTOR ADDRESS, AND PRIORITY LEVEL OF THE RH70/RP04. IF THE DEFAULT VAULE OF THE BUS ADDRESS DOES NOT RESPOND (TIMES OUT) TO ADDRESSING IT IS REPORTED AS AN ERROR. AFTER THE ERROR IS REPORTED ONE OF TWO COURSES OF ACTION WILL BE TAKEN:

1. IF THERE IS A MONITOR -- RETURN TO THE MONITOR
2. IF THERE ISN'T A MONITOR -- ASK FOR NEW ADDRESSES

STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR TO CHANGE THE CONTROL AND STATUS REGISTER #1 (RHCS1) ADDRESS, THE VECTOR ADDRESS, AND THE PRIORITY LEVEL.

4.4.2.1 ADDRESS SELECTION EXAMPLES

EXAMPLE #1

RHCS1=177200/176300..

EXAMPLE #2

RHCS1=177200/176300,
RHVEC=254/260,
RHPRIO=5/6..

EXAMPLE #3

RHCS1=177200,
RHVEC=254/260..

EXAMPLE #4

RH70/RP04 FAILED TO RESPOND TO ADDRESSING
RHCS1 ERR PC
177200 XXXXXX
RHCS1=177200/176300..

EXAMPLE #5

RHCS1=177200/1776\67\6300,
RHVEC=254,
RHPRIO=5,
RHCS1=176300..

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
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

4.4.3 DRIVE AND PARAMETER SELECTION

STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC SELECTION OF THE DRIVES TO TEST AND THE TESTS TO RUN.

STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR TO SELECT THE DRIVE(S) TO BE TESTED, THE TESTS TO BE EXECUTED, AND THE PARAMETERS TO USE.

4.4.3.1 DRIVE AND PARAMETER SELECTION DESCRIPTION

THE FOLLOWING IS A TABLE OF TERMS USED BY THE PSI.

"R"	REPEATS (ITERATIONS)
"FC"	FIRST CYLINDER ADDRESS
"LC"	LAST CYLINDER ADDRESS
"IC"	INCREMENT CYLINDER
"FT"	FIRST TRACK ADDRESS
"LT"	LAST TRACK ADDRESS
"IT"	INCREMENT TRACK
"FS"	FIRST SECTOR ADDRESS
"LS"	LAST SECTOR ADDRESS
"PAT"	PATTERN (USED FOR DATA TEST)
"WDX"	WORD OF PATTERN 0 WHERE X IS 1 TO 16
*"S"	ALL SEEK TESTS
*"T"	ALL TIMING TESTS
*"A"	ALL ADDRESS TESTS
*"D"	THE DATA TESTS
*"E"	THE EXERCISER

* USED BY USER AS INPUT.

NOTE1: ALL OTHERS ARE OUTPUT ONLY.

NOTE2: ALL NUMBERS WILL BE IN DECIMAL EXCEPT FOR THE PATTERN (PAT) AND WORDS (WDX) SELECTION. "PAT" WILL BE SELECTED BY A BIT (I.E. 001000(8)=PATTERN 9) AND "WDX" WILL BE IN OCTAL.

SPECIAL CASES OF CONTROL CHARACTERS

IF <..> IS TYPED WHILE A TEST IS OPEN FOR MODIFICATION (</>) AND OTHER TESTS IN THE "TEST COMMAND" STRING ARE TO BE OPENED, THEY WILL RECIEVE THE PARAMETERS OF THE TEST THAT IS OPEN WHEN <..> IS TYPED.

<.> IS ILLEGAL AS A "TEST" LINE TERMINATOR UNLESS A TEST IS TO BE OPENED.

ON STARTING THE PROGRAM ALL TESTS WILL RUN AGAINST ALL DRIVES
IN A WORST CASE MANNER. BUT IF THE USER DESIRES TO SELECT THE
DRIVES TO BE TESTED, THE TESTS TO BE EXECUTED, AND THE PARAMETERS
TO BE USED HE ENTERS CONVERSATION MODE BY TYPING CONTROL C (↑C).

THE PROGRAM WILL RESPOND WITH:

DRIVE(S)=

WE WILL ASSUME FOR OUR EXAMPLE THAT THE OPERATOR WISHES TO TEST
DRIVE #3 USING TESTS 2 THRU 7 AND TEST 11 AND DOES NOT DESIRE TO CHANGE
THE PARAMETERS (INITIAL CYLINDER ADDRESS, FINAL CYLINDER ADDRESS, ETC.).
THE USER WOULD TYPE "3<,>" WHICH SAYS "THIS IS THE END OF ANY
CHANGES TO THIS LINE BUT TAKE ME DEEPER FOR MORE CHANGES", AND <CR> "TAKE IT."

FOR CLARITY THE LINE WILL BE REPEATED AS IT LOOKS AFTER USER RESPONSE WITH
THE LINE BELOW IT ADDED, THE PROGRAMS RESPONSE:

DRIVE(S)=3,<CR>
TEST=

NOW THE USER INSERTS THE TEST NUMBERS HE DESIRES. IN THIS CASE
HE WANTS TESTS 2 THRU 7 AND TEST 11 SO HE TYPES 2-7<,> (TO
SEPERATE FIELD), 11<..> (END OF ANY CHANGES START EXECUTION),
<CR> (TAKE IT).

IT NOW LOOKS LIKE THIS

DRIVE(S)=3,<CR>
TEST=2-7,11.<CR>

IN OUR NEXT EXAMPLE WE WILL ASSUME THE USER WISHES TO TEST DRIVE
4 AND RUN TESTS 1 AND 3 THRU 11 WITH THE PARAMETER FOR TESTS 3
AND 10 MODIFIED.

THIS IS HOW HE WOULD RESPOND:

DRIVE(S)=4,<CR>
TEST=

THE USER NOW WILL INSERT A LINE TO GET TESTS 1 AND 3 THRU 11
BUT REMEMBER HE WISHES TO "OPEN" TESTS 3 AND 10 FOR PARAMETER
CHANGES. THE ENTIRE ENTRY IS SHOWN BELOW:

DRIVE(S)=4,<CR>
TEST=1,3/4-7,10/11.<CR>

NOTICE THIS SAYS SELECT TEST ONE, CONTINUE<,>; SELECT TEST 3, OPEN</>;
SELECT TESTS 4-7, CONTINUE<,>; SELECT TEST 10, OPEN</>; SELECT TEST
11, END OF INPUT <..>.

339
340
341
342
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
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387

398
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437

THE PSI SCANS THE TEST STRING AND DETERMINES THAT TEST #1'S
PARAMETERS WILL REMAIN THE SAME BUT
TEST 3'S MUST BE CHANGED. THEREFORE, TEST
3 IS OPENED FOR CHANGE.

RESPONDS: <FOR CLARITY ENTIRE TRANSACTION REPEATED>

```
DRIVE(S)=4,<CR>
TEST=1,3/4-7,10/11.<CR>
TEST 3
R=X ;WHERE X IS ITERATION
```

AS IN THE NORMAL MANNER TYPING A </> WILL OPEN THIS LINE FOR
MODIFICATION OF THE ITERATION COUNT, ENDING THE LINE WITH A
<.> (PERIOD) WOULD TERMINATE THE CHANGES FOR THIS TEST, AND AS
IN THE EXAMPLE SHOWN USING A <,> (COMMA) WILL GET US THE NEXT
PARAMETER.

```
DRIVE(S)=4,<CR>
TEST=1,3/4-7,10/11,<CR>
TEST 3
R=1,<CR> ;DO NOT ALTER-BUT CONTINUE
FC=N ;WHERE N IS FIRST CYLINDER ADDRESS
```

THE USER DOES NOT WISH TO CHANGE "FC" SO THE FOLLOWING ACTION IS TAKEN:

```
DRIVE(S)=4,<CR>
TEST=1,3/4-7,10/11.<CR>
TEST 3
R=1,<CR> ;DO NOT ALTER THIS LINE BUT CONTINUE
FC=0,<CR> ;DO NOT ALTER THIS LINE BUT CONTINUE
LC=410
```

THE PROGRAM RESPONDS WITH THE PREVIOUSLY ASSIGNED PARAMETER FOR LAST
CYLINDER ADDRESS IN THIS CASE USING 410 AS THE EXAMPLE. THIS IS
WHAT THE USER INTENDED TO MODIFY AND IS WHY HE OPENED
TEST #3. HE WANTS TO CHANGE IT TO CYLINDER 20. THEREFORE
HE TYPES </> (OPEN LINE FOR CHANGE), "20" (THE NEW VALUE),
<.> (THIS IS THE END OF CHANGES FOR THIS DRIVE), <CR>
(TAKE IT).

THE TOTAL TRANSACTION AND RESPONSE:

```
DRIVE(S)=4,<CR>
TEST=1,3/4-7,10/11.<CR>
TEST 3
R= 1,<CR>
FC=0,<CR>
LC= 410/20.<CR>
TEST 10
R=1
```


438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459

THE PROGRAM HAS LOADED TEST #3 WITH ITS NEW PARAMETERS. TESTS NUMBER 4 THRU 7 WITH THEIR PREVIOUSLY ASSIGNED PARAMETERS AND IS NOW WAITING FOR CHANGES TO TEST 10.

THE USER TYPES </> (OPEN THIS LINE FOR CHANGE), "10" (MAKE ITERATION COUNT 10), <.> (THIS IS THE END OF CHANGES TO THIS TEST) <CR> (TAKE IT).

THE PROGRAM NOW LOADS TEST 10 WITH THE NEW PARAMETERS, TEST 11 WITH PREVIOUSLY ASSIGNED PARAMETERS AND RESPONDS WITH:

DRIVE(S)=

SINCE THE USER DID NOT END THE CONVERSATION MODE WITH A <...> THE PROGRAM HAS LOOPED BACK TO THE BEGINNING LOOKING FOR MORE CHANGES. THAT IS TO SAY, AFTER THE ENTRY FOR DRIVE SELECTION A <,><CR> WILL CAUSE THE TEST MESSAGE TO BE REPEATED AND FURTHER CHANGES CAN BE MADE. HOWEVER, AT SOME POINT IN ORDER TO EXECUTE THE CHANGES MADE WE MUST NOTIFY THE PROGRAM WE ARE FINISHED BY TYPING <...>.


```

460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511

```

EXAMPLE #1

DRIVE=4.<CR> ;SELECT DRIVE #4, TERMINATE AND
;BEGIN EXECUTION USING PREVIOUSLY ASSIGNED
;PARAMETERS

EXAMPLE #2

DRIVE=0.<CR> ;SELECT DRIVE #0 AND MAKE CHANGES ""
TEST=1-5.<CR> ;RUN TEST 1 THRU 5 ONLY, USE DEFAULT
;PARAMETERS AND TERMINATE AND EXECUTE ""

EXAMPLE #3

DRIVE=2.<CR> ;SELECT DRIVE #2 AND MAKE CHANGES ""
TEST=1-5,6/7/10/<CR> ;RUN TEST 1-5 WITH DEFAULT PARAMETERS, OPEN
TEST 6 ;TEST 6, 7 AND 10 FOR CHANGES
R=1.<CR> ;LEAVE R AS IS MOVE TO NEXT PARAMETER
FC=0/10.<CR> ;SET "FC" CYLINDER ADDRESS TO 10 "" END CHANGES
;TO TEST #6

TEST 7
R=1/50.<CR> ;50 ITERATIONS, MOVE TO NEXT PARAMETER
FC=0.<CR> ;DO NOT CHANGE "FC" CYLINDER ADDRESS BUT CONTINUE "",
LC=410/50.<CR> ;TEST 10 IS STILL PENDING
;AND WILL THEREFORE BE
;SET TO THE SAME PARAMETERS
;AS TEST 7 -- START EXECUTION

EXAMPLE #4

DRIVE=0.<CR> ;SELECT DRIVE #0 AND MAKE CHANGES
TEST=S,E.<CR> ;RUN ALL SEEK TESTS AND THE EXERCISER

EXAMPLE #5

DRIVE=1.<CR>
TEST=S/D.<CR> ;RUN ALL SEEK TESTS (OPEN FOR CHANGES) AND
TEST 0 ;THE DATA TEST (WITH DEFAULT PARAMETERS).
R=10.<CR> ;RUN WITH 10 ITERATIONS
FC=0/10.<CR> ;CHANGE FIRST CYLINDER ADDRESS
;AND START EXECUTION
;TESTS 1-6 WILL BE ASSIGNED THE
;SAME PARAMETERS AS TEST 0

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

EXAMPLE #6

```

DRIVE=1,<CR>
TEST=S/,<CR>
TEST 0
  R=10/100.<CR>
TEST 1
  R=100/1000.<CR>
TEST 2
  R=1/10,<CR>
  FC=0/50,<CR>
  LC=410/51.<CR>
TEST 3
  R=1.<CR>
TEST 4
  R=1..<CR>

```

;OPEN THE SEEK TESTS (TST 0-6)
 ;CHANGE TO 100 ITERATIONS; <.> MOVE TO NEXT TEST
 ;CHANGE R TO 1000 ITERATIONS; <.> MOVE TO NEXT TEST
 ;CHANGE R TO 10 ITERATIONS; <.> MOVE TO NEXT LINE
 ;CHANGE FC TO 50; <.> MOVE TO NEXT LINE
 ;CHANGE LC TO 51; <.> MOVE TO NEXT TEST
 ;<.> MOVE TO NEXT TEST
 ;USE TESTS 4'S PARAMETERS
 ;FOR TEST 5 & 6 START EXECUTION

EXAMPLE #7

```

DRIVE=1,<CR>
TEST=D/,<CR>
TEST 15
  R=1/1000,<CR>
  FC=0/10,<CR>
  LC=410/10,<CR>
  IC=64/0,<CR>
  FT=0/2,<CR>
  LT=18/2,<CR>
  IT=1,<CR>
  FS=0/4,<CR>
  LS=22/4,<CR>
  PAT=177777/400..<CR>

```

;SELECT AND OPEN THE DATA TEST
 ;DO 1000 ITERATION OF TEST PATTERN
 ;#8 ON CYLINDER 10, TRACK 2, SECTOR 4
 ;RUN WITH PATTERN #8

577
576
575
574
573
572
571
570
569
568
567
566
565
564
563
562
561
560
559
558
557
556
555
554
553
552
551
550
549
548
547
546
545
544
543
542
541
540
539
538
537
536
535
534
533
532
531
530
529
528
527
526
525
524
523
522
521
520
519
518
517
516
515
514
513
512
511
510
509
508
507
506
505
504
503
502
501
500
499
498
497
496
495
494
493
492
491
490
489
488
487
486
485
484
483
482
481
480
479
478
477
476
475
474
473
472
471
470
469
468
467
466
465
464
463
462
461
460
459
458
457
456
455
454
453
452
451
450
449
448
447
446
445
444
443
442
441
440
439
438
437
436
435
434
433
432
431
430
429
428
427
426
425
424
423
422
421
420
419
418
417
416
415
414
413
412
411
410
409
408
407
406
405
404
403
402
401
400
399
398
397
396
395
394
393
392
391
390
389
388
387
386
385
384
383
382
381
380
379
378
377
376
375
374
373
372
371
370
369
368
367
366
365
364
363
362
361
360
359
358
357
356
355
354
353
352
351
350
349
348
347
346
345
344
343
342
341
340
339
338
337
336
335
334
333
332
331
330
329
328
327
326
325
324
323
322
321
320
319
318
317
316
315
314
313
312
311
310
309
308
307
306
305
304
303
302
301
300
299
298
297
296
295
294
293
292
291
290
289
288
287
286
285
284
283
282
281
280
279
278
277
276
275
274
273
272
271
270
269
268
267
266
265
264
263
262
261
260
259
258
257
256
255
254
253
252
251
250
249
248
247
246
245
244
243
242
241
240
239
238
237
236
235
234
233
232
231
230
229
228
227
226
225
224
223
222
221
220
219
218
217
216
215
214
213
212
211
210
209
208
207
206
205
204
203
202
201
200
199
198
197
196
195
194
193
192
191
190
189
188
187
186
185
184
183
182
181
180
179
178
177
176
175
174
173
172
171
170
169
168
167
166
165
164
163
162
161
160
159
158
157
156
155
154
153
152
151
150
149
148
147
146
145
144
143
142
141
140
139
138
137
136
135
134
133
132
131
130
129
128
127
126
125
124
123
122
121
120
119
118
117
116
115
114
113
112
111
110
109
108
107
106
105
104
103
102
101
100
99
98
97
96
95
94
93
92
91
90
89
88
87
86
85
84
83
82
81
80
79
78
77
76
75
74
73
72
71
70
69
68
67
66
65
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0

EXAMPLE #8

```

DRIVE=1, <CR>
TEST=D/, <CR>
TEST IS
R=1000, <CR>
FC=10, <CR>
LC=10, <CR>
IC=0, <CR>
FT=2, <CR>
LT=2, <CR>
IT=1, <CR>
FS=4, <CR>
LS=4, <CR>
PAT=000400/401, <CR>
WD1=165555/125252, <CR>
WD2=133333/52525.. <CR>
; <...> START EXECUTION

```

;USE THE SAME PARAMETERS AS IN EXAMPLE #7 BUT ALSO SPECIFY YOUR OWN DATA PATTERN (PAT #0).
;RUN WITH PATTERN #8 & #0 (0=OPERATOR INPUT)
;FIRST WORD OF PATTERN 0
;SECOND WORD OF PATTERN 0

EXAMPLE #9

```

DRIVE=0,1,4,
TEST=0-5/
TEST 0
R=10,
FC=0,
LC=410/1..

```

;TEST DRIVES 0,1, AND 4 IN SEQUENCE
;CHANGE TEST 0-5
;CHANGE LAST CYLINDER FROM 410
;TO 1 FOR TEST #0-5; START TESTING

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
6255. OPERATING PROCEDURE
-----5.1 OPERATIONAL SWITCH SETTINGS

WITH SW<15:0>=0 THE PROGRAM WILL PRINT OUT ON
ERRORS AND CONTINUE IN TEST.
THE SWITCH SETTINGS ARE:

SW<15>=1...HALT ON ERROR
SW<14>=1...LOOP ON TEST
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<11>=1...INHIBIT ITERATIONS
SW<10>=1...RING BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<08>=1...PRINT ERROR MESSAGE ON LINE PRINTER
SW<07>=1...READ CONTROL SWITCH SETTINGS FROM TTY
SW<06>=1...INHIBIT TIME REPORTS (TESTS 7-12)
SW<05>=1...REPORT ONE ERROR PER SECTOR (TESTS 13&14)
SW<04>=1...INHIBIT WRITES (TEST 15)
SW<03>=1...INHIBIT WRITE CHECKS (TEST 15)
SW<02>=1...INHIBIT READ AND SOFTWARE COMPARES (TEST 15)
SW<01>=1...INHIBIT SOFTWARE COMPARES (TEST 15)
SW<00>=1...PERFORM READ AFTER WRITE CHECK ERROR (TEST 15)

5.2 SUBROUTINE ABSTRACTS

THE SUBROUTINE CALLS USED BY THIS PROGRAM ARE:

1. IOT (USED FOR SCOPE)
THIS CALL IS PLACED BETWEEN EACH TEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH TEST IN LOCATIONS "SLPADR" AND "SLPERR" AS IT IS BEING ENTERED.
2. EMT (USED FOR ERROR)
THIS CALL IS USED TO REPORT ALL ERRORS (REFER TO 6).
3. TRAP (USED FOR TYPE, TYPOC, TYPOS, TYPON, TYPDS, RDCHR, RDLIN, SAVREG, AND RESREG)
4. JSR R,ADR

THE FOLLOWING SECTION CONTAINS AN ABSTRACT
FOR EACH SUBROUTINE

6. ERRORS

THERE ARE A NUMBER OF ERRORS THAT CAN OCCUR IN THIS PROGRAM. WHEN AN ERROR IS ENCOUNTERED THE CALL TO THE ERROR ROUTINE IS MADE AND IF SW<13> IS NOT SET AN ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TYPE OUT WILL CONTAIN THE FOLLOWING:

1. AN ERROR MESSAGE
2. A DATA HEADER
3. A DATA STRING

REFER TO THE FOLLOWING SECTION FOR THE DIFFERENT ERRORS THAT CAN OCCUR.

6.1 ERROR TYPES

THE ERRORS THAT OCCUR IN THIS PROGRAM FALL INTO THREE (3) CATEGORIES DEFINED AND EXPLAINED AS FOLLOWS:

6.1.1 DRIVER ERROR

THESE ERRORS WILL BE DETECTED BY THE RM70/RP04 DRIVER. THERE ARE TWO CLASSES OF DRIVER ERRORS; THOSE THAT CAN NOT BE IDENTIFIED IN A MANNER THAT ALLOWS THE INFORMATION TO BE RETURNED TO A "DATA PARAMETER BLOCK" (DPB) AND THOSE THAT CAN. THE FIRST CLASS WILL BE REPORTED BY ERROR CALLS (EMT'S) 1-5 WITHIN THE DRIVER. THE SECOND CLASS WILL PASS THE ERROR CODES TO THE STATUS/ERROR WORD (DPB+16) OF THE PROPER DPB.

6.1.2 NON-FATAL ERROR

THESE ERRORS WILL BE DUE TO "DISK" OR "DATA" FAILURES WHICH WILL BE REPORTED AS THEY OCCUR. AFTER REPORTING THE ERROR THE PROGRAM WILL CONTINUE TESTING.

6.1.3 FATAL ERROR

THIS TYPE OF ERROR WILL BE THE RESULT OF ANY KIND OF ERROR THAT INHIBITS THE PROGRAM FROM TESTING THE DISK.

THIS ERROR WILL BE REPORTED WHEN IT OCCURS, THEN THE PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

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
7196.2 ERROR RECOVERY

6.2.1 PRETEST ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THEN DEPENDING ON HOW THE PROGRAM WAS STARTED IT WILL ASK FOR THE DRIVES AND ADDRESSES FOR TESTING OR RETURN TO MONITOR.

6.2.2 NON-FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED AND THE PROGRAM WILL CONTINUE IN TEST.

6.2.3 FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THE PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

7. RESTRICTIONS

BEFORE STARTING THE PROGRAM THE OPERATOR MUST INSURE THAT A FORMATTED PACK IS LOADED AND WRITE ENABLED IN THE DRIVE(S) TO BE TESTED.

8. MISCELLANEOUS
-----8.1 EXECUTION TIME

TO MAKE ONE PASS OF THE PROGRAM TAKES APPROXIMATELY 4 MINUTES FOR EVERY 4K OF CORE. THE RUN TIME INCREASES LINEARLY FOR EACH DRIVE TESTED.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100 AND EXTENDS DOWN IN MEMORY.

8.3 COMMON TAGS

THE COMMON TAGS USED IN THIS PROGRAM START AT 1100 AND EXTEND UP IN MEMORY.

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

9.4 TIMING TESTS (TST7-TST12) PRINTOUTS

AT THE COMPLETION OF EACH OF THE TIMING TESTS THE TIME OF THE MINIMUM SEEK, MAXIMUM SEEK, AND THE AVERAGE OF ALL OF THE SEEKS PERFORMED ARE TYPED ON THE TTY. THE NUMBER OF SEEKS THAT HAD TIMES BELOW THE MINIMUM TIME ALLOWED WILL BE TYPED ON THE SAME LINE AS THE MINIMUM TIME. THE NUMBER ABOVE THE MAXIMUM WILL BE TYPED ON THE SAME LINE AS THE MAXIMUM TIME, AND THE TOTAL NUMBER OF SEEKS PERFORMED WILL BE ON THE SAME LINE AS THE AVERAGE.

NOTE: THE PROGRAM STALLS FOR 2 MILLISECONDS BETWEEN SEEK ORDERS. THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES. THE 2 MILLISECOND STALL BETWEEN SEEK ORDERS IS SPECIFIED BY THE RPO4 VENDOR. THE SEEK TIMES SPECIFIED BY THE VENDOR ARE POSITIONER MOVEMENT TIMES ONLY AND ARE NOT A MEASUREMENT OF EFFECTIVE SEEK TIME.

8.4.1 TIMING TOLERANCES

1. TEST 7 -- ROTATIONAL SPEED TIMES

60 HZ
MINIMUM=16340 US
MAXIMUM=17000 US
NOMINAL=16670 US

50 HZ
MINIMUM=16250 US
MAXIMUM=17090 US
NOMINAL=16670 US

2. TEST 10 -- ONE CYLINDER SEEK TIMES

MINIMUM=3000 US
MAXIMUM=10000 US
NOMINAL=7000 US

3. TEST 11 -- AVERAGE SEEK TIMES

MINIMUM=26000 US
MAXIMUM=30000 US
NOMINAL=28000 US

4. TEST 12 -- MAXIMUM SEEK TIMES

MINIMUM=46000 US
MAXIMUM=52000 US
NOMINAL=50000 US

771
772
773
774
775
776
777
778
779
790
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

EXAMPLE #1

ROTATIONAL SPEED TIMES
MIN=16670 US
MAX=16690 US
AVG=16680 US 10 SEEKS TIMED

ONE CYLINDER SEEK TIMES
** 0, 1, 2, ..., 410 **
MIN=5350 US
MAX=6920 US
AVG=5550 US 409 SEEKS TIMED
** 410, 409, 408, ..., 0 **
MIN=5140 US
MAX=5960 US
AVG=5430 US 410 SEEKS TIMED

AVERAGE SEEK TIMES
** 0 TO 136 **
MIN=27770 US
MAX=28640 US
AVG=28230 US 128 SEEKS TIMED
** 136 TO 0 **
MIN=27990 US
MAX=28550 US
AVG=28220 US 128 SEEKS TIMED

MAXIMUM SEEK TIMES
** 0 TO 410 **
MIN=49990 US
MAX=51980 US
AVG=51010 US 128 SEEKS TIMED
** 410 TO 0 **
MIN=48120 US
MAX=50650 US
AVG=49340 US 128 SEEKS TIMED

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

ROTATIONAL SPEED TIMES
MIN=16670 US
MAX=16690 US
AVG=16680 US 10 SEEKS TIMED

ONE CYLINDER SEEK TIMES
** 0, 1, 2,...410 **
MIN=5470 US
MAX=10940 US 3 ABOVE THE MAXIMUM OF 10000 US
AVG=5830 US 409 SEEKS TIMED
** 410, 409, 408,...0 **
MIN=5040 US
MAX=5970 US
AVG=5330 US 410 SEEKS TIMED

AVERAGE SEEK TIMES
** 0 TO 136 **
MIN=29730 US
MAX=31620 US 73 ABOVE THE MAXIMUM OF 30000 US
AVG=30320 US 128 SEEKS TIMED
** 136 TO 0 **
MIN=28620
MAX=31230 US 128 ABOVE THE MAXIMUM OF 30000 US
AVG=30800 US 128 SEEKS TIMED

MAXIMUM SEEK TIMES
** 0 TO 410 **
MIN=53510 US
MAX=54240 US 128 ABOVE THE MAXIMUM OF 52000 US
AVG=54020 US 128 SEEKS TIMED
** 410 TO 0 **
MIN=52050 US
MAX=54550 US 128 ABOVE THE MAXIMUM OF 52000 US
AVG=52210 US 128 SEEKS TIMED

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

 WITH ALL SWITCHES ON A "0" AN "END OF PASS" MESSAGE WILL BE
 TYPED AT THE COMPLETION OF TESTING A DRIVE AND THE "END OF TEST"
 TYPEOUT WILL OCCUR WHEN ALL DRIVES HAVE BEEN TESTED.

8.6 RH70/RP04 DRIVER

ALL OF THE COMMUNICATION WITH THE RH70/RP04 IS PERFORMED
 VIA A COMMON DRIVER. THIS SECTION CONTAINS THE USERS GUIDE
 FOR THE DRIVER.

8.6.1. TO INITIALIZE THE DRIVER:

```
JSR PC,RPINIT
RETURN
```

UPON RETURN YOU MUST EXAMINE THE "DRVSTA" TABLE TO DETERMINE
 THE DRIVES THAT ARE ONLINE FOR TESTING. THE DRVSTA TABLE IS
 EIGHT BYTES; ONE BYTE PER DRIVE. THE STATE OF EACH DRIVE WILL
 BE INDICATED AS FOLLOWS:

```
>0 ON LINE
=0 OFFLINE
<0 UNSAFE OR NONEXISTENT
```

WITH NON-RP04 DRIVES INDICATED AS OFFLINE. AND THE DRIVE TYPE
 WILL BE INDICATED IN THE "DRVTYP" TABLE. USING "DRVSTA" AND
 "DRVTYP" THE FOLLOWING CAN BE DETERMINED:

ONLINE	OFFLINE	UNSAFE	NONEXISTENT
-----	-----	-----	-----
DRVSTA>0	DRVSTA=0	DRVSTA<0	DRVSTA<0
DRVTYP=(RHDT)	DRVTYP=(RHDT)	DRVTYP=(RHDT)	DRVTYP=0

THE RPINIT ROUTINE WILL DO A READIN PRESET AND WILL SET FMT22.

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

FOR YOU.

CALL:

JSR RO,RPO4
PNTDPB
RETURN1
RETURN2

;MAKE THE CALL
;ADDRESS OF DPB*
;RETURN IF QUEUE IS FULL
;RETURN IF REQUEST IS IN
;QUEUE OR THERE IS AN
;ERROR CONDITION

*DPB (DATA PARAMETER BLOCK)

PNTDPB: .BYTE 0
.BYTE 0
.BYTE 0
.BYTE 0
.WORD 0
.WORD 0

.BYTE 0
.BYTE 0

.WORD 0
.WORD 0

.WORD 0

;(0) DRIVE NUMBER
;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
;(2) COMMAND
;(3) PSEL AND A17 AND A16
;(4) WORD COUNT (MUST BE NEG.)
;(6) BUFFER ADDRESS OR
REGISTER TABLE POINTER
;(10) SECTOR ADDRESS OR
FIRST REG. INDEX
;(11) TRACK ADDRESS OR
LAST REG. INDEX
;(12) CYLINDER ADDRESS
;(14) ERROR TABLE POINTER
POINTS TO THE FIRST OF TWENTY
LOCATIONS OF WHERE THE DRIVER
IS TO STORE THE RH70/RPO4
REGISTERS ON AN ERROR. IF LEFT
ZERO REGISTERS ARE NOT SAVED.
;(16) STATUS/ERROR INDICATOR
BIT15=1=>ERROR OCCURRED
BIT07=1=>DONE
BIT14-BIT09 AND BIT06-BIT03
INDICATE TYPE OF ERROR

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

TO UTILIZE THIS CAPABILITY YOU MUST SUPPLY THE "RP TIMER" ROUTINE WITH THE ELAPSED TIME IN THE FOLLOWING MANNER:

```

MOV    #16.,-(SP)    ;16 MILLISECONDS BETWEEN
                        ;CLOCK TICKS
JSR    RO,RPTMR      ;CALL THE TIMER ROUTINE
  
```

IT SHOULD BE NOTED THAT YOU MUST PROVIDE THE CODE TO DRIVE THE CLOCK. AND THE ELAPSED TIME MUST BE IN MILLISECONDS. THE DRIVER WILL SET THE TIMEOUT TO 1 SECOND FOR ALL POSITIONING AND DATA TRANSFER OPERATIONS AND WILL SET THE TIMEOUT TO 30 SECONDS FOR ERROR RECOVERY OPERATIONS.

8.6.4. EXAMPLE - WRITE 1000. WORDS

```

1$:   JSR    RO,RPO3    ;CALL THE DRIVER
      WRTDPB ;DPB ADDRESS
      BR     1$        ;WAIT FOR QUEUE IF FULL
2$:   TST    WRTDPB+16 ;WAIT FOR COMMAND TO COMPLETE
      BEQ    2$
      BMI    ERROR1   ;ERROR OCCURRED
      .
      .
      .
  
```

```

WRTDPB: .BYTE 5      ;DRIVE #5
        .BYTE 0
        .BYTE 161   ;WRITE COMMAND
        .BYTE 0
        .WORD -1000. ;WORD COUNT
        .WORD WRTBUF ;BUFFER ADDRESS
        .BYTE 3     ;SECTOR
        .BYTE 5     ;TRACK
        .WORD 400   ;CYLINDER
        .WORD ERRTB5 ;ERROR TABLE
        .WORD 0     ;STATUS/ERROR INDICATOR
  
```

ATERNATE DPB SETUP

```

WRTDPB: .WORD 5      ;THIS SETUP ACHIEVED
        .WORD WRITE ;EVERYTHING THE
        .WORD -1000. ;ABOVE TABLE DID, BUT
        .WORD WRTBUF ;IN A CLEANER FORMAT
        .BYTE 3,5
        .WORD 400,ERRTB5,0
  
```

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
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028

<u>MNEMONIC</u>	<u>INDEX</u>
RHCS1	0
RHWC	2
RHBA	4
RHDA	6
RHCS2	10
RHDS1	12
RHER1	14
RHAS	16
RHLA	20
RHDB	22
RHMR	24
RHDT	26
RHSN	30
RHOF	32
RHCA	34
RHCC	36
RHER2	40
RHER3	42
RHEC1	44
RHEC2	46
RHBAE	50
RHCS3	52

8.6.6. COMMANDS PERFORMED BY THE DRIVER

<u>COMMAND</u>	<u>CODE</u>	<u>COMMAND TYPE</u>
NO OPERATION	101	N
UNLOAD	103	N
SEEK	105	P
RECALIRATE	107	P
DRIVE CLEAR	111	N
RELEASE	113	N
OFFSET	115	P
RETURN TO CENTER	117	P
READIN PRESET	121	N
PACK ACKNOWLEDGE	123	N
SEARCH	131	P

1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079

SET FORMAT	143	S
SELECT DRIVE	145	S
WRITE CHECK DATA	151	D
WRITE CHECK HEADER AND DATA	153	D
WRITE DATA	161	D
WRITE HEADER & DATA	163	D
READ DATA	171	D
READ HEADER & DATA	173	D

N = HOUSEKEEPING
P = POSITIONING
D = DATA TRANSFER
S = SPECIAL PROVIDED BY THE DRIVER

8.6.7. DPB STATUS/ERROR INDICATOR

THIS INDICATOR WILL INFORM THE USER OF THE RESULTS OF THE REQUEST.
THIS IS ACCOMPLISHED BY SETTING VARIES BITS OF THE INDICATOR TO
A ONE.

<u>BIT NO.</u>	<u>MEANING IF ON A "1"</u>
15	ERROR OCCURRED DONE (BIT07=0); BITS 14-10 SPECIFIES TYPE DONE (BIT07=1); BITS 06-03 SPECIFIES TYPE
14(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON AN OFFLINE OR UNSAFE DRIVE
13(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON A DRIVE THAT HAS AN UNLOAD REQUEST IN QUEUE.
12(2)	PERSISTENT UNSAFE CONDITION EXIST.
11(2)	PARITY ERROR OCCURRED
10(2)(4)	FATAL PARITY ERROR. A MASSBUS CLEAR WAS PERFORMED, ALL QUEUES WERE EMPTIED, AND ALL DRVACT'S SET TO THE IDLE STATE

1080	9(3)(4)	SOFTWARE TIMEOUT OCCURRED ON THIS DRIVE
1081		
1082	8(4)	SOFTWARE TIMEOUT OCCURRED ON ANOTHER DRIVE
1083		
1084	7	DONE
1085		
1086	6(2)	ERROR OCCURRED DURING AN I/O OPERATION
1087		
1088	5(2)	ERROR OCCURRED DURING AN OPERATION OTHER THAN I/O.
1089		
1090		
1091	4(2)	UNSAFE CONDITION OCCURRED
1092		
1093	3(2)	DRIVE ERROR OCCURRED THAT CAUSED AN AUTOMATIC "RECALIBRATE" SEQUENCE
1094		
1095		
1096	(1) =>	REQUEST WASN'T PUT IN QUEUE. (RH70/RP04 REGISTERS WERE NOT SAVED)
1097		
1098		
1099	(2) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A "DRIVE CLEAR" TO THE DRIVE. NOTE: ALL RH70/RP04 REGISTERS ARE SAVED AS PER DPB+14 BEFORE THE "DRIVE CLEAR".
1100		
1101		
1102		
1103		
1104	(3) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A MASSBUS INIT. ALL RH70/RP04 REGISTERS FOR THE DRIVE WERE SAVED AS PER DPB+14 BEFORE THE INIT.
1105		
1106		
1107		
1108		
1109		
1110	(4) =>	A "RECALIBRATE" SHOULD BE ISSUED BEFORE ANY OTHER COMMAND.
1111		

1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160

THERE ARE A FEW ERRORS THAT CAN OCCUR THAT CAN NOT BE INDICATED IN A DPB.

WHEN THIS TYPE OF ERROR IS DETECTED BY THE DRIVER IT WILL MAKE AN ERROR CALL OF THE FORM "ERROR N", WHERE "N" IS THE ERROR NUMBER AND THE ERROR WILL BE AN EMT INSTRUCTION.

N	TYPE	DATA AVAILABLE
-	----	-----
1	ILLEGAL SUBSYSTEM INTERRUPT OCCURRED (SC=0)	*R4= RHCS1'S ADDRESS
2	ILLEGAL DRIVE INTERRUPT OCCURRED	R1= DRIVE NUMBER R3= ATA BIT *R4= RHCS1'S ADDRESS R5= (RHAS) RPERRS =RHDS1 RPERRS+2=RHER1 RPERRS+4=RHER2 RPERRS+6=RHER3
3	MASSBUS CONTROL BUS PARITY ERROR OCCURRED(MCPE=1)	RD.ADR= ADDRESS OF REG. READ RD.WRD= WORD READ
4	CONTROL BUS PARITY ERROR OCCURRED(PAR=1)	WRT.AD= ADDRESS OF REG. WRITTEN WRT.WD= WORD WRITTEN RD.WRD= WORD READ BACK
5	ATTENTION FROM AN OFFLINE, NONEXISTENT OR PREVIOUSLY DETECT UNSAFE DRIVE OCCURRED. NOTE: COULD HAVE BEEN CAUSED BY A BIT PICK OR DROP WHEN READING RHAS.	R1= DRIVE NUMBER R3= ATA BIT *R4= RHCS1'S ADDRESS R5= (RHAS) RPERRS =RHDS1 RPERRS+2=RHER1 RPERRS+4=RHER2 RPERRS+6=RHER3
6	RESERVED	
7.	RESERVED	

* THIS IS THE ACTUAL UNIBUS ADDRESS (177200)

9. PROGRAM DESCRIPTION

THIS PROGRAM CONTAINS SIXTEEN TESTS NUMBERED 0-17 IN OCTAL. TESTS 0-6 WILL READ THE CYLINDER, TRACK, AND SECTOR INFORMATION FROM THE HEADER, USING A "READ HEADER AND DATA" COMMAND, AND THEN CHECK THE INFORMATION FOR VALIDITY, THUS, INSURING THE SEEK OPERATION FUNCTIONS PROPERLY. TEST 7-12 WILL MEASURE THE ROTATIONAL SPEED, THE ONE CYLINDER SEEK, THE AVERAGE SEEK, AND THE MAXIMUM SEEK TIMES TO INSURE THEY ARE ALL WITHIN THE TOLERANCES ALLOWED. TEST 13 AND 14 INSURES THE SECTOR AND TRACK ADDRESSING CIRCUITRY WORKS PROPERLY. TEST 15 VERIFIES THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONAL. AND TEST 16 WILL STRESS AND CHECK THE READ/WRITE AND SERVO SYSTEMS. TEST 17 IS SIMILAR TO TEST 15 EXCEPT FOR THE FOLLOWING
1) ONLY LAST TWO CYLINDERS USED. 2) ONLY EIGHT PATTERNS USED.
3) ALL OF AVAILABLE MEMORY IS USED.

THE PROGRAM WILL START BY IDENTIFYING ITSELF AND DETERMINING ALL DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH THE LOWEST NUMERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER ALL OF THE DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE (TEST 0-15) WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE. THE DRIVE TO BE TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS, AN "END OF PASS" MESSAGE WILL BE TYPED AT THE COMPLETION OF EACH PASS, AND AN "END OF TEST" MESSAGE WILL BE TYPED AFTER TESTING ALL DRIVES.

REFER TO THE FOLLOWING SECTION FOR A DETAILED DESCRIPTION OF EACH TEST.

1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DECDOC VER 00.06 15-SEP-75 14:04 PAGE 01

1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

DOCUMENT

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"

COPYRIGHT 1975
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DECDOC VER 00.06 15-SEP-75 14:04 PAGE 02

TABLE OF CONTENTS

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
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

13	CONTROL SWITCH SETTINGS
35	OPERATIONAL SWITCH SETTINGS
57	TRAP CATCHER
64	STARTING ADDRESSES
76	BASIC DEFINITIONS
181	MEMORY MANAGEMENT DEFINITIONS
246	COMMON TAGS
1048	ERROR POINTER TABLE
1353	START OF PROGRAM
1570	#### TESTS ####
1605	*** SEEK TESTS (0-6) ***
1919	*** TIMING TESTS (7-12) ***
2309	*** ADDRESSING TESTS (13-14) ***
2450	*** DATA TEST (15) ***
2605	*** EXERCISE TEST (16) ***
2694	*** DATA TEST WITH ALL MEMORY (17) ***
2894	END OF PASS ROUTINE
2953	*** SUBROUTINES ***
2956	ERROR HANDLER ROUTINE
3005	TYPERR - TYPE ERROR ROUTINE
3301	TYPE ROUTINE
3348	BINARY TO OCTAL (ASCII) AND TYPE

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

TABLE OF CONTENTS

3426	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3494	TTY INPUT ROUTINE
3624	SCOPE HANDLER ROUTINE
3680	SAVE AND RESTORE RO-RS ROUTINES
3726	TRAP DECODER
3741	TRAP TABLE
3761	DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
3801	SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE
3819	DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
3882	TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
3906	INTEGER DIVIDE ROUTINE
3988	RANDOM NUMBER GENERATOR ROUTINE
4035	ROUTINE TO SIZE MEMORY
4111	LP.AVL - LINE PRINTER AVAILABLE
4131	ST.CLK - CLOCK STARTUP ROUTINE
4201	LDCMD - LOAD COMMAND
4219	CALL.A - CALL RPO4 DRIVER USING "DPB.A"
4250	CALL.B - CALL RPO4 DRIVER USING "DPB.B"
4291	CALL.C - CALL RPO4 DRIVER USING "DPB.C"
4331	DRVCAL - DRIVER (RPO4) CALL USING "DTADPB"
4431	ERINDX - FORM ERROR INDEX
4483	STALL - USED TO STALL AFTER A DRIVE FUNCTION IS COMPLETE

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

TABLE OF CONTENTS

4512	TWOMS - STALL FOR 2 MS BETWEEN SEEKS IN TESTS 10- - 12
4534	VERIFY - VALIDATE HEADER ON IMPLIED SEEKS
4562	SRCH00 - INITIALIZE THE DRIVE FOR THE TIMING TESTS
4605	DORTI - RETURN FROM INTERRUPT
4611	STRMTR - START THE TIMERS
4631	COUNT - THIS ROUTINE COUNTS THE ELAPSED TIME
4670	TYPTIM - TYPE TIMES
4755	INCTRK - INCREMENT TRACK NUMBER
4774	INCCYL - INCRMENT CYLINDER NUMBER
4792	FILBUF - FILL BUFFER WITH ADDRESS DATA
4814	CLRBUF - CLEAR BUFFER
4839	CKSCTR - CHECK SECTOR DATA
4933	SETBUF - SET BUFFER TO DATA PATTERN
4969	DATCMP - DATA COMPARE
5063	FILRAN - FILL DATA BUFFER WITH RANDOM PATTERN
5084	RANCK - RANDOM PATTERN BUFFER CHECK
5145	RANPAT - RANDOM PATTERN GENERATOR
5166	RANADR - RANDOM ADDRESS GENERATOR
5221	GETSWR - GET THE CONTROL SWITCH SETTINGS
5246	GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
5310	GETNUM - ROUTINE TO GET A NUMBER
5369	GT.PRM - GET PARAMETERS

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

TABLE OF CONTENTS

5724	CK.OCT -- CHECK FOR OCTAL CHARACTER
5744	CK.DEC -- CHECK FOR DECIMAL CHARACTER
5764	CK.CHR -- CHECK CHARACTER
5798	CK.DIG - CHECK DIGIT
5864	CK.NUM - CHECK NUMBER (OCTAL)
5913	RH11/RP04 DRIVER (REV. 0.9)
7211	ASCIZ MESSAGES
7297	ERROR HEADER (EM) MESSAGES
7373	STATUS/ERROR INDICATOR MESSAGES
7433	DATA HEADER (DT) MESSAGES
7559	DATA TABLE (DT)
7614	DATA FORMAT (DF) TABLE

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DECDOC VER 00.06 15-SEP-75 14:04 PAGE 06

3 COPYRIGHT (C) 1974, 1975
DIGITAL EQUIPMENT CORP.
MAYNARD, MASS. 01754

PROGRAM BY J. LACEY/C. HESS/SUB MALLICK

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
PACKAGE (MAINDEC-11-DZQAC-A2).

13

CONTROL SWITCH SETTINGS

14

SWITCH	STATE	USE
15	0	WRITE PACK BEFORE TESTING (TEST 16)
	1	INHIBIT WRITING PACK BEFORE TESTING (TES
14	0	NO STALL BETWEEN DRIVE FUNCTIONS
	1	STALL AFTER EVERY DRIVE FUNCTION
13	0	USE SPECIFIC STALL TIME
	1	USE RANDOM STALL
12	0	NO INCREMENTING STALL IN TEST 4
	1	DO INCREMENTING STALL IN TEST 4
8	0	DO IMPLIED SEEKS WITH DATA TRANSFERS
	1	DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
7	0	DO "READ HEADER AND DATA" IN TESTS 0-6
	1	DO EXPLICIT SEEKS IN TESTS 0-6
6	0	60 HZ
	1	50 HZ
5	0	RUN WATCHDOG TIMER
	1	INHIBIT WATCHDOG TIMER

35

OPERATIONAL SWITCH SETTINGS

36

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	PRINT ERROR MESSAGE ON LINE PRINTER
7	READ CONTROL SWITCH SETTINGS FROM TTY
6	INHIBIT TIME REPORTS (TESTS 7-12)
5	REPORT ONE ERROR PER SECTOR (TESTS 13 &
4	INHIBIT WRITES (TEST 15)

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

J03

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DERPKC.P11

MACY11 27(732) 14-OCT-76 10:36 PAGE 36

1486
1487
1488

3
2

INHIBIT WRITE CHECKS (TEST 15)
INHIBIT READ AND SOFTWARE COMPARES (TEST

1488

1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531

1
0

INHIBIT SOFTWARE COMPARES (TEST 15)
PERFORM READ AFTER WRITE CHECK ERROR (TE

57

TRAP CATCHER

60 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

64

STARTING ADDRESSES

66 200 = NORMAL START

68 204 = SELECT OPERATING PARAMETERS

70 210 = SELECT RH11/RP04 ADDRESSES

72 214 = COMBINATION OF 204 AND 210

76

BASIC DEFINITIONS

78 INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

89 GENERAL PURPOSE REGISTER DEFINITIONS

101 PRIORITY LEVEL DEFINITIONS

111 "SWITCH REGISTER" SWITCH DEFINITIONS

139 DATA BIT DEFINITIONS (BIT00 TO BIT15)

167.. BASIC "CPU" TRAP VECTOR ADDRESSES

1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DECD0C VER 00.06 15-SEP-75 14:04 PAGE 08

181

MEMORY MANAGEMENT DEFINITIONS

183 KT11 VECTOR ADDRESS

187 KT11 STATUS REGISTER ADDRESSES

194 KERNAL "I" PAGE DESCRIPTOR REGISTERS

205 KERNAL "I" PAGE ADDRESS REGISTERS

244 *****

%
.ENABLE ABS,AMA
.TITLE MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
.*COPYRIGHT (C) 1974,1975
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY J. LACEY/C. HESS/SUB MALLICK
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZGAC-A2).
.*

.SBTTL CONTROL SWITCH SETTINGS

```

.*
.* SWITCH STATE USE
.* -----
.* 15 0 WRITE PACK BEFORE TESTING (TEST 16)
.* 1 1 INHIBIT WRITING PACK BEFORE TESTING (TEST 16)
.* 14 0 NO STALL BETWEEN DRIVE FUNCTIONS
.* 1 1 STALL AFTER EVERY DRIVE FUNCTION
.* 13 0 USE SPECIFIC STALL TIME
.* 1 1 USE RANDOM STALL
.* 12 0 NO INCREMENTING STALL IN TEST 4
.* 1 1 DO INCREMENTING STALL IN TEST 4
.* 8 0 DO IMPLIED SEEKS WITH DATA TRANSFERS
.* 1 1 DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
.* 7 0 DO "READ HEADER AND DATA" IN TESTS 0-6
.* 1 1 DO EXPLICIT SEEKS IN TESTS 0-6
.* 6 0 60 HZ
.* 1 1 50 HZ
.* 5 0 RUN WATCHDOG TIMER
.* 1 1 INHIBIT WATCHDOG TIMER

```

.SBTTL OPERATIONAL SWITCH SETTINGS

.*

	SWITCH	USE
1588	*	
1589	*	
1590	*	15 HALT ON ERROR
1591	*	14 LOOP ON TEST
1592	*	13 INHIBIT ERROR TYPEOUTS
1593	*	11 INHIBIT ITERATIONS
1594	*	10 BELL ON ERROR
1595	*	9 LOOP ON ERROR
1596	*	8 PRINT ERROR MESSAGE ON LINE PRINTER
1597	*	7 READ CONTROL SWITCH SETTINGS FROM TTY
1598	*	6 INHIBIT TIME REPORTS (TESTS 7-12)
1599	*	5 REPORT ONE ERROR PER SECTOR (TESTS 13 & 14)
1600	*	4 INHIBIT WRITES (TEST 15)
1601	*	3 INHIBIT WRITE CHECKS (TEST 15)
1602	*	2 INHIBIT READ AND SOFTWARE COMPARES (TEST 15)
1603	*	1 INHIBIT SOFTWARE COMPARES (TEST 15)
1604	*	0 PERFORM READ AFTER WRITE CHECK ERROR (TEST 15)
1605	*	
1606	*	
1607	*	

.SBTTL TRAP CATCHER

1610 000000 .=0
 1611 ; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
 1612 ; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
 1613 ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

.SBTTL STARTING ADDRESSES

1615 000200 .=200
 1616 ; *200 = NORMAL START
 1617 JMP @#START1
 1618 000200 000137 004154 ; *204 = SELECT OPERATING PARAMETERS
 1619 JMP @#START2
 1620 000204 000137 004244 ; *210 = SELECT RH11/RP04 ADDRESSES
 1621 JMP @#START3
 1622 000210 000137 004144 ; *214 = COMBINATION OF 204 AND 210
 1623 JMP @#START4
 1624 000214 000137 004234

.SBTTL BASIC DEFINITIONS

1628 ; *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
 1629 STACK= 1100
 1630 001100 .EQUIV EMT,ERROR ; ;BASIC DEFINITION OF ERROR CALL
 1631 .EQUIV IOT,SCOPE ; ;BASIC DEFINITION OF SCOPE CALL
 1632 PS= 177776 ; ;PROCESSOR STATUS WORD
 1633 177776 .EQUIV PS,PSW
 1634 STKLMT= 177774 ; ;STACK LIMIT REGISTER
 1635 177774 PIRQ= 177772 ; ;PROGRAM INTERRUPT REQUEST REGISTER
 1636 177772 SWR= 177570 ; ;SWITCH REGISTER
 1637 177570 DISPLAY=SWR

.SBTTL GENERAL PURPOSE REGISTER DEFINITIONS

1640 R0= %0 ; ;GENERAL REGISTER
 1641 000000 R1= %1 ; ;GENERAL REGISTER
 1642 000001 R2= %2 ; ;GENERAL REGISTER
 1643 000002

1644	000003	R3=	%3	::	GENERAL REGISTER
1645	000004	R4=	%4	::	GENERAL REGISTER
1646	000005	R5=	%5	::	GENERAL REGISTER
1647	000006	R6=	%6	::	GENERAL REGISTER
1648	000007	R7=	%7	::	GENERAL REGISTER
1649		.EQUIV	R6, SP	::	STACK POINTER
1650		.EQUIV	R7, PC	::	PROGRAM COUNTER
1651					
1652		:*PRIORITY LEVEL DEFINITIONS			
1653	000000	PR0=	0	::	PRIORITY LEVEL 0
1654	000040	PR1=	40	::	PRIORITY LEVEL 1
1655	000100	PR2=	100	::	PRIORITY LEVEL 2
1656	000140	PR3=	140	::	PRIORITY LEVEL 3
1657	000200	PR4=	200	::	PRIORITY LEVEL 4
1658	000240	PR5=	240	::	PRIORITY LEVEL 5
1659	000300	PR6=	300	::	PRIORITY LEVEL 6
1660	000340	PR7=	340	::	PRIORITY LEVEL 7
1661					
1662		:*"SWITCH REGISTER" SWITCH DEFINITIONS			
1663	100000	SW15=	100000		
1664	040000	SW14=	40000		
1665	020000	SW13=	20000		
1666	010000	SW12=	10000		
1667	004000	SW11=	4000		
1668	002000	SW10=	2000		
1669	001000	SW09=	1000		
1670	000400	SW08=	400		
1671	000200	SW07=	200		
1672	000100	SW06=	100		
1673	000040	SW05=	40		
1674	000020	SW04=	20		
1675	000010	SW03=	10		
1676	000004	SW02=	4		
1677	000002	SW01=	2		
1678	000001	SW00=	1		
1679		.EQUIV	SW09, SW9		
1680		.EQUIV	SW08, SW8		
1681		.EQUIV	SW07, SW7		
1682		.EQUIV	SW06, SW6		
1683		.EQUIV	SW05, SW5		
1684		.EQUIV	SW04, SW4		
1685		.EQUIV	SW03, SW3		
1686		.EQUIV	SW02, SW2		
1687		.EQUIV	SW01, SW1		
1688		.EQUIV	SW00, SW0		
1689					
1690		:*DATA BIT DEFINITIONS (BIT00 TO BIT15)			
1691	100000	BIT15=	100000		
1692	040000	BIT14=	40000		
1693	020000	BIT13=	20000		
1694	010000	BIT12=	10000		
1695	004000	BIT11=	4000		
1696	002000	BIT10=	2000		
1697	001000	BIT09=	1000		
1698	000400	BIT08=	400		
1699	000200	BIT07=	200		

1700	000100	BIT06=	100
1701	000040	BIT05=	40
1702	000020	BIT04=	20
1703	000010	BIT03=	10
1704	000004	BIT02=	4
1705	000002	BIT01=	2
1706	000001	BIT00=	1
1707		.EQUIV	BIT09, BIT9
1708		.EQUIV	BIT08, BIT8
1709		.EQUIV	BIT07, BIT7
1710		.EQUIV	BIT06, BIT6
1711		.EQUIV	BIT05, BIT5
1712		.EQUIV	BIT04, BIT4
1713		.EQUIV	BIT03, BIT3
1714		.EQUIV	BIT02, BIT2
1715		.EQUIV	BIT01, BIT1
1716		.EQUIV	BIT00, BIT0

1718		.*BASIC "CPU" TRAP VECTOR ADDRESSES	
1719	000004	ERRVEC= 4	:: TIME OUT AND OTHER ERRORS
1720	000010	RESVEC= 10	:: RESERVED AND ILLEGAL INSTRUCTIONS
1721	000014	TBITVEC=14	:: "T" BIT
1722	000014	TRTVEC= 14	:: TRACE TRAP
1723	000014	BPTVEC= 14	:: BREAKPOINT TRAP (BPT)
1724	000020	IOTVEC= 20	:: INPUT/OUTPUT TRAP (IOT) **SCOPE**
1725	000024	PWRVEC= 24	:: POWER FAIL
1726	000030	EMTVEC= 30	:: EMULATOR TRAP (EMT) **ERROR**
1727	000034	TRAPVEC=34	:: "TRAP" TRAP
1728	000060	TKVEC= 60	:: TTY KEYBOARD VECTOR
1729	000064	TPVEC= 64	:: TTY PRINTER VECTOR
1730	000240	PIRQVEC=240	:: PROGRAM INTERRUPT REQUEST VECTOR


```

1731
1732      .SBTTL MEMORY MANAGEMENT DEFINITIONS
1733
1734      ;*KT11 VECTOR ADDRESS
1735
1736      MMVEC= 250
1737
1738      ;*KT11 STATUS REGISTER ADDRESSES
1739
1740      SR0= 177572
1741      SR1= 177574
1742      SR2= 177576
1743      SR3= 172516
1744
1745      ;*KERNAL "I" PAGE DESCRIPTOR REGISTERS
1746
1747      KIPDR0= 172300
1748      KIPDR1= 172302
1749      KIPDR2= 172304
1750      KIPDR3= 172306
1751      KIPDR4= 172310
1752      KIPDR5= 172312
1753      KIPDR6= 172314
1754      KIPDR7= 172316
1755
1756      ;*KERNAL "I" PAGE ADDRESS REGISTERS
1757
1758      KIPAR0= 172340
1759      KIPAR1= 172342
1760      KIPAR2= 172344
1761      KIPAR3= 172346
1762      KIPAR4= 172350
1763      KIPAR5= 172352
1764      KIPAR6= 172354
1765      KIPAR7= 172356
1766

```

```

1767 ;OP CODE DEFINITIONS
1768 000101 NOOP=101
1769 000103 UNLOAD=103
1770 000105 SEEK=105
1771 000107 RECAL=107
1772 000111 DRVCLR=111
1773 000113 RELEASE=113
1774 000115 OFFSET=115
1775 000117 RTC=117
1776 000121 READIN=121
1777 000123 PACK=123
1778 000131 SEARCH=131
1779 000151 WRCKD=151
1780 000153 WRCKHD=153
1781 000161 WRITE=161
1782 000163 WRTHD=163
1783 000171 READ=171
1784 000173 READHD=173
1785 000141 GETREG=141
1786 000143 SETFORM=143
1787 000145 SELDRV=145

```

```

1788 ;OTHER EQUATES
1789
1790
1791 165000 TRCKWC = -(256.*22.)
1792 177400 SCTRWC = -256.
1793 010000 FMT22=10000
1794

```

```

;WORD COUNT FOR TRACK
;WORD COUNT FOR SECTOR
;FORMAT 22 BIT

```



```

1795 ;*****
1796
1797 .SBTTL COMMON TAGS
1798
1799 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1800 ;*USED IN THE PROGRAM.
1801
1802 000046 000046 .=46
1603 000046 015210 $ENDAD ;;LOGICAL END OF PROGRAM
1804
1805 001100 .=1100
1806
1807 001100 SCMTAG: ;;START OF COMMON TAGS
1808 001100 000000 $PASS: .WORD 0 ;;CONTAINS PASS COUNT
1809 001102 000 $STSTM: .BYTE 00 ;;CONTAINS THE TEST NUMBER
1810 001103 000 $SERFLG: .BYTE 00 ;;CONTAINS ERROR FLAG
1811 001104 000000 $SICNT: .WORD 00 ;;CONTAINS SUBTEST ITERATION COUNT
1812 001106 000000 $SLPADR: .WORD 00 ;;CONTAINS SCOPE LOOP
1813 001110 000000 $SLPERR: .WORD 00 ;;CONTAINS SCOPE RETURN FOR ERRORS
1814 001112 000000 $SERTTL: .WORD 00 ;;CONTAINS TOTAL ERRORS DETECTED
1815 001114 000 $SITEMB: .BYTE 0 ;;CONTAINS ITEM CONTROL BYTE
1816 001115 001 $SERMAX: .BYTE 1 ;;CONTAINS MAX. ERRORS PER TEST
1817 001116 000000 $SERRPC: .WORD 00 ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1818 001120 000000 $SGADR: .WORD 00 ;;CONTAINS OF 'GOOD' DATA
1819 001122 000000 $SBDADR: .WORD 00 ;;CONTAINS OF 'BAD' DATA
1820 001124 000000 $SGODAT: .WORD 00 ;;CONTAINS 'GOOD' DATA
1821 001126 000000 $SBDDAT: .WORD 00 ;;CONTAINS 'BAD' DATA
1822 001130 000000 000000 000000 .WORD 0,0,0 ;;RESERVED--NOT TO BE USED
1823 001136 177560 $TKS: 177560 ;;TTY KBD STATUS
1824 001140 177562 $TKB: 177562 ;;TTY KBD BUFFER
1825 001142 177564 $TPS: 177564 ;;TTY PRINTER STATUS REG.
1826 001144 177566 $TPB: 177566 ;;TTY PRINTER BUFFER REG.
1827 001146 000 $SNLL: .BYTE 0 ;;CONTAINS NULL CHARACTER FOR FILLS
1828 001147 002 $SFILLS: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1829 001150 012 $SFILLC: .BYTE 12 ;;INSERT FILL CHARS. AFTER A "LINE FEED"
1830 001151 000 $STPFLG: .BYTE 0 ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1831 001152 000000 $SREGAD: .WORD 0 ;;CONTAINS THE FROM
1832 WHICH ($SREGO) WAS OBTAINED
1833 001154 000000 $SREGO: .WORD 0 ;;CONTAINS (($SREGAD)+0)
1834 001156 000000 $SREG1: .WORD 00 ;;CONTAINS (($SREGAD)+2)
1835 001160 000000 $SREG2: .WORD 00 ;;CONTAINS (($SREGAD)+4)
1836 001162 000000 $SREG3: .WORD 00 ;;CONTAINS (($SREGAD)+6)
1837 001164 000000 $SREG4: .WORD 00 ;;CONTAINS (($SREGAD)+10)
1838 001166 000000 $SREG5: .WORD 00 ;;CONTAINS (($SREGAD)+12)
1839 001170 000000 $STMP0: .WORD 00 ;;USER DEFINED
1840 001172 000000 $STMP1: .WORD 00 ;;USER DEFINED
1841 001174 000000 $STMP2: .WORD 00 ;;USER DEFINED
1842 001176 000000 $STIMES: 0 ;;MAX. NUMBER OF ITERATIONS
1843 001200 000000 $SESCAPE: 0 ;;ESCAPE ON ERROR
1844 001202 177607 000377 $SBELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
1845 001206 077 $SQUES: .ASCII /?/ ;;QUESTION MARK
1846 001207 015 $SCRLF: .ASCII <15> ;;CARRIAGE RETURN
1847 001210 000012 $SLF: .ASCIZ <12> ;;LINE FEED
1848 001212 000000 $BUF: 0
1849 001214 000000 $TEMP1: 0
1850 001216 000000 $TEMP2: 0

```

1851	001220	000000		TEMP3:	0		
1852	001222	000000		TEMP4:	0		
1853	001224	000000		TEMP5:	0		
1854	001226	000000		TEMP6:	0		
1855	001230	000000		TEMP7:	0		
1856	001232	000000		TEMP9:	0		
1857	001234	000000		TEMP10:	0	;0=70	1=45
1858	001236	177740		LERADD:	177740		
1859	001240	177742		HERADD:	177742		
1860	001242	177744		MEMERR:	177744		
1861	001244	000001		RPT17:	1		
1862	001246	000000		C.SWR:	.WORD	0	:CONTROL SWITCHES
1863	001250	000000		CNTRLC:	.WORD	0	:CONTROL "C" FLAG
1864	001252	000000		BUSADR:	.WORD	0	:GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
1865	001254	000000		LPTAVL:	.WORD	0	:LPT AVAILABLE STATUS (0=NO,1=YES)
1866	001256	000000		DRVSEL:	.WORD	0	:DRIVES SELECTED FOR TESTING
1867	001260	037777		TSTNMS:	.WORD	37777	:RUN TESTS 0-15
1868	001262	000000		CLKSTA:	.WORD	0	:CLOCK STATUS (0=NO CLOCK,+1=KW11-P, AND -1=KW11-L)
1869							:16 MILLISECONDS PER CLOCK TICK
1870	001264	000000		TICKMS:	.WORD	↑D16	:16666 MICROSECONDS PER CLOCK TICK
1871	001266	040432		TICKUS:	.WORD	↑D16666	
1872	001270	000000		BYPASS:	.WORD	0	
1873	001272	000000		CHKDRV:	.WORD	0	:DRIVE UNDER TEST
1874	001274	000000		DRVMSK:	.WORD	0	:DRIVE MASK BIT
1875	001276	000000		SVSTAT:	.WORD	0	:STATUS/ERROR INDICATOR IS SAVED HERE ON AN ERROR
1876							:CYLINDER READ
1877	001300	000000		CYL.RD:	.WORD	0	:TRACK READ
1878	001302	000000		TRK.RD:	.WORD	0	:SECTOR READ
1879	001304	000000		SEC.RD:	.WORD	0	:CYLINDER DESIRED
1880	001306	000000		CYL.DS:	.WORD	0	:SECTOR DESIRED
1881	001310	000000		SEC.DS:	.WORD	0	:TRACK DESIRED
1882	001312	000000		TRK.DS:	.WORD	0	:MINIMUM TIME
1883	001314	000000		TIM.UP:	.WORD	0	:NUMBER OF COUNTS BELOW MIN. LIMIT
1884	001316	000000			.WORD	0	:MAXIMUM TIME
1885	001320	000000			.WORD	0	:NUMBER OF COUNTS ABOVE MAX. LIMIT
1886	001322	000000			.WORD	0	:TOTAL TIME OF ALL SEEKS
1887	001324	000000	000000		.WORD	0,0	:NUMBER OF SEEKS PERFORMED
1888	001330	000000		TIM.DN:	.WORD	0	:MINIMUM TIME
1889	001332	000000			.WORD	0	:NUMBER OF COUNTS BELOW MIN. LIMIT
1890	001334	000000			.WORD	0	:MAXIMUM TIME
1891	001336	000000			.WORD	0	:NUMBER OF COUNTS ABOVE MAX. LIMIT
1892	001340	000000			.WORD	0	:TOTAL TIME OF ALL SEEKS
1893	001342	000000	000000		.WORD	0,0	:NUMBER OF SEEKS PERFORMED
1894	001346	000000			.WORD	0	:POINTS TO TABLE OF TIMES
1895	001350	000000		TIM.PT:	.WORD	0	:FATAL WRITE CHECK ERROR FLAG (TST 15)
1896	001352	000000		WCEFLG:	.WORD	0	:VARIABLE STALL (TEST 4)
1897	001354	000000		STALLO:	.WORD	0	:SAVE DISK ADDRESS (TST16)
1898	001356	000000	000000	SVADR:	.WORD	0,0	
1899							
1900				:CONSTANTS STORAGE			
1901	001362	176700		RH.ADR:	.WORD	176700	:RH11/RP04 CONTROL AND STATUS REG. #1
1902	001364	000254	000240	RHVEC:	.WORD	254,5*32.	:RH11/RP04 VECTOR AND PRIORITY
1903	001370	000104	000106	PKV:	.WORD	104,106	:KW11-P VECTOR ADDRESS
1904	001374	172540		PKCS:	.WORD	172540	:KW11-P CONTROL AND STATUS REG.
1905	001376	172542		PKB:	.WORD	172542	:KW11-P COUNT SET BUFFER
1906	001400	172544		PKC:	.WORD	172544	:KW11-P COUNTER

1907 001402 000100 000102
 1908 001406 177546
 1909 001410 177564
 1910 001412 177566
 1911 001414 177514
 1912 001416 177516
 1913 001420 000012
 1914 001422 000012
 1915 001424 000031
 1916 001426 006
 1917
 1918 001427 000
 1919
 1920
 1921 001430 000001
 1922 001432 000002
 1923 001434 000004
 1924 001436 000010
 1925 001440 000020
 1926 001442 000040
 1927 001444 000100
 1928 001446 000200
 1929 001450 000400
 1930 001452 001000
 1931 001454 002000
 1932 001456 004000
 1933 001460 010000
 1934 001462 020000
 1935 001464 040000
 1936 001466 100000
 1937
 1938
 1939
 1940 001470 000000
 1941 001472 000000
 1942 001474 003000
 1943 001476 000000
 1944 001500 000000
 1945 001502 000000
 1946 001504 000000
 1947 001506 000000
 1948 001510 000000
 1949 001512 000000
 1950 001514 000000
 1951
 1952
 1953 001516 002116
 1954 001520 002130
 1955 001522 002152
 1956 001524 002170
 1957 001526 002206
 1958 001530 002224
 1959 001532 002242
 1960 001534 002260
 1961 001536 002272
 1962 001540 002276

LKV: .WORD 100,102
 LKS: .WORD 177546
 TPS: .WORD 177564
 TPB: .WORD 177566
 LPS: .WORD 177514
 LPB: .WORD 177516
 STALL1: .WORD ↑D10
 STALL2: .WORD ↑D10
 MXSTAL: .WORD ↑D25
 ERR.CT: .BYTE 6

;KW11-L VECTOR ADDRESS
 ;KW11-L STATUS REGISTER
 ;TTY PRINTER STATUS
 ;TTY PRINTER BUFFER
 ;LINE PRINTER STATUS
 ;LINE PRINTER BUFFER
 ;10 MILLISECONDS STALL (TEST 0-6)
 ;10 MILLISECONDS STALL (TEST 13-16)
 ;MAX. INCREMENTING STALL ALLOWED IN TEST 4
 ;NUMBER OF ERRORS ALLOWED IN TESTS 13-16
 ;BEFORE GOING TO THE NEXT TEST
 ;RESERVED

;BIT TABLE
 BITS: .WORD BIT00
 .WORD BIT01
 .WORD BIT02
 .WORD BIT03
 .WORD BIT04
 .WORD BIT05
 .WORD BIT06
 .WORD BIT07
 .WORD BIT08
 .WORD BIT09
 .WORD BIT10
 .WORD BIT11
 .WORD BIT12
 .WORD BIT13
 .WORD BIT14
 .WORD BIT15

;COMMON STORAGE FOR TEST PARAMETER

PRM: .WORD 0
 RPT: .WORD 0
 FC: .WORD 0
 LC: .WORD 0
 IC: .WORD 0
 FT: .WORD 0
 LT: .WORD 0
 IT: .WORD 0
 FS: .WORD 0
 LS: .WORD 0
 PAT: .WORD 0

;REPEAT COUNTS FOR ALL TESTS
 ;FIRST CYLINDER
 ;LAST CYLINDER
 ;INCREMENT CYLINDER
 ;FIRST TRACK
 ;LAST TRACK
 ;INCREMENT TRACK
 ;FIRST SECTOR
 ;LAST SECTOR
 ;PATTERN CODE

;TABLE OF PARAMETER POINTERS

PRMPT: .WORD PRM0
 .WORD PRM1
 .WORD PRM2
 .WORD PRM3
 .WORD PRM4
 .WORD PRM5
 .WORD PRM6
 .WORD PRM7
 .WORD PRM10
 .WORD PRM11

1963 001542 002302
1964 001544 002306
1965 001546 002316
1966 001550 002326
1967 001552 002354
1968
1969

.WORD PRM12
.WORD PRM13
.WORD PRM14
.WORD PRM15
.WORD PRM16

:PARAMETER UPPER LIMIT
PRMLMT: .WORD ↑D32767
.WORD ↑D410
.WORD ↑D410
.WORD ↑D410
.WORD ↑D18
.WORD ↑D18
.WORD ↑D18
.WORD ↑D21
.WORD ↑D21
.WORD 177777

:"R"
:"FC"
:"LC"
:"IC"
:"FT"
:"LT"
:"IT"
:"FS"
:"LS"
:"PAT"

1970 001554 077777
1971 001556 000632
1972 001560 000632
1973 001562 000632
1974 001564 000022
1975 001566 000022
1976 001570 000022
1977 001572 000025
1978 001574 000025
1979 001576 177777
1980

:TABLE OF MESSAGE POINTERS
PRMMSG: .WORD MSG.R
.WORD MSG.FC
.WORD MSG.LC
.WORD MSG.IC
.WORD MSG.FT
.WORD MSG.LT
.WORD MSG.IT
.WORD MSG.FS
.WORD MSG.LS

1981
1982 001600 040006
1983 001602 040010
1984 001604 040013
1985 001606 040016
1986 001610 040021
1987 001612 040024
1988 001614 040027
1989 001616 040032
1990 001620 040035
1991

:STATUS/ERROR INDICATOR MESSAGES POINTER TABLE
STATBL: .WORD MSGB14 ;OFFLINE OR UNSAFE DRIVE REQUESTED
.WORD MSGB13 ;UNLOAD DRIVE REQUESTED
.WORD MSGB12 ;PERSISTENT UNSAFE
.WORD MSGB11 ;PARITY ERROR OCCURRED
.WORD MSGB10 ;FATAL PARITY ERROR
.WORD MSGB09 ;SOFTWARE TIMEOUT ON THIS DRIVE
.WORD MSGB08 ;SOFTWARE TIMEOUT ON ANOTHER DRIVE
.WORD MSGB06 ;ERROR OCCURRED DURING I/O OPERATION
.WORD MSGB05 ;ERROR OCCURRED DURING NON-I/O OPERATION
.WORD MSGB04 ;UNSAFE OCCURRED
.WORD MSGB03 ;AUTOMATIC RECALIBRATE SEQUENCE OCCURRED

1992
1993 001622 041431
1994 001624 041473
1995 001626 041524
1996 001630 041546
1997 001632 041574
1998 001634 041617
1999 001636 041656
2000 001640 041720
2001 001642 041764
2002 001644 042034
2003 001646 042054
2004

:DEFAULT VALUES OF TESTS PARAMETERS
DFLT: .WORD 225,10.,410.,0,0 ;RECAL/SEEK (T0)
.WORD 677,100.,0,128.,0,0,0,0 ;SEEK/SEEK (T1)
.WORD 237,1,0,410.,1,0,0 ;INCREMENT SEEK (T2)
.WORD 237,1,0,256.,1,0,0 ;STEPPING SEEK (T3)
.WORD 237,1,0,410.,1,0,0 ;OSCILLATING SEEK (T4)

2005
2006 001650 000225 000012 000632
2007 001656 000000 000000
2008 001662 000677 000144 000000
2009 001670 000200 000000 000000
2010 001676 000000 000000 000000
2011 001704 000237 000001 000000
2012 001712 000632 000001 000000
2013 001720 000000
2014 001722 000237 000001 000000
2015 001730 000400 000001 000000
2016 001736 000000
2017 001740 000237 000001 000000
2018 001746 000632 000001 000000


```

2019 001754 000000
2020 001756 000237 000001 000000 .WORD 237,1,0,410.,1,0,0 ;CONVERGING/DIVERGING SEEK (T5)
2021 001764 000632 000001 000000
2022 001772 000000
2023 001774 000237 000001 000000 .WORD 237,1,0,410.,1,0,0 ;SERVO ADDRESSING LOGIC NOISE (T6)
2024 002002 000632 000001 000000
2025 002010 000000
2026 002012 000223 000001 000000 .WORD 223,1,0,0,0 ;ROTATIONAL SPEED TIMING TEST (T7)
2027 002020 000000 000000
2028 002024 000001 000001 .WORD 1,1 ;ONE CYLINDER SEEK TIMING TEST (T10)
2029 002030 000001 000001 .WORD 1,1 ;AVERAGE SEEK TIMING TEST (T11)
2030 002034 000001 000001 .WORD 1,1 ;MAXIMUM SEEK TIMEING TEST (T12)
2031 002040 000023 000001 000000 .WORD 23,1,0,0 ;SECTOR ADDRESSING TEST (T13)
2032 002046 000000
2033 002050 000203 000001 000000 .WORD 203,1,0,0 ;TRACK ADDRESSING TEST (T14)
2034 002056 000000
2035 002060 001777 000001 000000 .WORD 1777,1,0,410.,64.,0,18.,1,1,0,177777 ;DATA TEST (T15)
2036 002066 000632 000100 000000
2037 002074 000022 000001 000001
2038 002102 000000 177777
2039 002106 000007 047040 000000 .WORD 7,20000.,0,410. ;EXERCISER (T16)
2040 002114 000632

```

;PARAMETER TABLES

:RECAL/SEEK (T0)

```

PRM0: .WORD 225
RPT0: .WORD 10.
LC0: .WORD 410.
FT0: .WORD 0
FS0: .WORD 0

```

:SEEK/SEEK (T1)

```

PRM1: .WORD 677
RPT1: .WORD 100.
FC1: .WORD 0
LC1: .WORD 128.
IC1: .WORD 0
FT1: .WORD 0
LT1: .WORD 0
FS1: .WORD 0
LS1: .WORD 0

```

:INCREMENT SEEK (T2)

```

PRM2: .WORD 237
RPT2: .WORD 1
FC2: .WORD 0
LC2: .WORD 410.
IC2: .WORD 1
FT2: .WORD 0
FS2: .WORD 0

```

:STEPPING SEEK (T3)

```

PRM3: .WORD 237
RPT3: .WORD 1
FC3: .WORD 0

```

```

2041
2042
2043
2044
2045 002116 000225
2046 002120 000012
2047 002122 000632
2048 002124 000000
2049 002126 000000
2050
2051
2052 002130 000677
2053 002132 000144
2054 002134 000000
2055 002136 000200
2056 002140 000000
2057 002142 000000
2058 002144 000000
2059 002146 000000
2060 002150 000000
2061
2062
2063 002152 000237
2064 002154 000001
2065 002156 000000
2066 002160 000632
2067 002162 000001
2068 002164 000000
2069 002166 000000
2070
2071
2072 002170 000237
2073 002172 000001
2074 002174 000000

```

2075	002176	000400	LC3:	.WORD	256.
2076	002200	000001	IC3:	.WORD	1
2077	002202	000000	FT3:	.WORD	0
2078	002204	000000	FS3:	.WORD	0
2079					
2080			:OSCILLATING SEEK (T4)		
2081	002206	000237	PRM4:	.WORD	237
2082	002210	000001	RPT4:	.WORD	1
2083	002212	000000	FC4:	.WORD	0
2084	002214	000632	LC4:	.WORD	410.
2085	002216	000001	IC4:	.WORD	1
2086	002220	000000	FT4:	.WORD	0
2087	002222	000000	FS4:	.WORD	0
2088					
2089			:CONVERGING/DIVERGING SEEK (T5)		
2090	002224	000237	PRM5:	.WORD	237
2091	002226	000001	RPT5:	.WORD	1
2092	002230	000000	FC5:	.WORD	0
2093	002232	000632	LC5:	.WORD	410.
2094	002234	000001	IC5:	.WORD	1
2095	002236	000000	FT5:	.WORD	0
2096	002240	000000	FS5:	.WORD	0
2097					
2098			:SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)		
2099	002242	000237	PRM6:	.WORD	237
2100	002244	000001	RPT6:	.WORD	1
2101	002246	000000	FC6:	.WORD	0
2102	002250	000632	LC6:	.WORD	410.
2103	002252	000001	IC6:	.WORD	1
2104	002254	000000	FT6:	.WORD	0
2105	002256	000000	FS6:	.WORD	0
2106					
2107			:ROTATIONAL SPEED TIMING TEST (T7)		
2108	002260	000223	PRM7:	.WORD	223
2109	002262	000001	RPT7:	.WORD	1
2110	002264	000000	FC7:	.WORD	0
2111	002266	000000	FT7:	.WORD	0
2112	002270	000000	FS7:	.WORD	0
2113					
2114			:ONE CYLINDER SEEK TIMING TEST (T10)		
2115	002272	000001	PRM10:	.WORD	1
2116	002274	000001	RPT10:	.WORD	1
2117					
2118			:AVERAGE SEEK TIMING TEST (T11)		
2119	002276	000001	PRM11:	.WORD	1
2120	002300	000001	RPT11:	.WORD	1
2121					
2122			:MAXIMUM SEEK TIMING TEST (T12)		
2123	002302	000001	PRM12:	.WORD	1
2124	002304	000001	RPT12:	.WORD	1
2125					
2126			:SECTOR ADDRESSING TEST (T13)		
2127	002306	000023	PRM13:	.WORD	23
2128	002310	000001	RPT13:	.WORD	1
2129	002312	000000	FC13:	.WORD	0
2130	002314	000000	FT13:	.WORD	0

2131
 2132
 2133 002316 000203
 2134 002320 000001
 2135 002322 000000
 2136 002324 000000
 2137
 2138
 2139 002326 001777
 2140 002330 000001
 2141 002332 000000
 2142 002334 000632
 2143 002336 000100
 2144 002340 000000
 2145 002342 000022
 2146 002344 000001
 2147 002346 000001
 2148 002350 000000
 2149 002352 177777
 2150
 2151
 2152 002354 000007
 2153 002356 047040
 2154 002360 000000
 2155 002362 000632
 2156
 2157
 2158
 2159 002364 040126
 2160 002366 000000
 2161 002370 003142
 2162 002372 003244
 2163
 2164 002374 040126
 2165 002376 000000
 2166 002400 003131
 2167 002402 003255
 2168
 2169 002404 040160
 2170 002406 040241
 2171 002410 000454
 2172 002412 001750
 2173
 2174 002414 040274
 2175 002416 040342
 2176 002420 005050
 2177 002422 005670
 2178
 2179 002424 040363
 2180 002426 040431
 2181 002430 010770
 2182 002432 012120
 2183
 2184 002434 002474
 2185 002436 002534
 2186 002440 002574

: TRACK ADDRESSING TEST (T14)

PRM14: .WORD 203
 RPT14: .WORD 1
 FC14: .WORD 0
 FS14: .WORD 0

: DATA TEST (T15)

PRM15: .WORD 1777
 RPT15: .WORD 1
 FC15: .WORD 0
 LC15: .WORD 410.
 IC15: .WORD 64.
 FT15: .WORD 0
 LT15: .WORD 18.
 IT15: .WORD 1
 FS15: .WORD 1
 LS15: .WORD 0
 PTRN15: .WORD 177777

: EXERCISER (T16)

PRM16: .WORD 7
 RPT16: .WORD 20000.
 FC16: .WORD 0
 LC16: .WORD 410.

; SEEK TIMING LIMITS

T7A: .WORD MSG7
 .WORD 0
 .WORD 1634. ; (16.67-2%)*2
 .WORD 1700. ; (16.67+2%)*2
 T7B: .WORD MSG7
 .WORD 0
 .WORD 1625. ; (16.67-2.5%)*2
 .WORD 1709. ; (16.67+2.5%)*2
 T10: .WORD MSG10A
 .WORD MSG10B
 .WORD 300. ; (7-4)*2
 .WORD 1000. ; (7+3)*2
 T11: .WORD MSG11A
 .WORD MSG11B
 .WORD 2600. ; (28-2)*2
 .WORD 3000. ; (28+2)*2
 T12: .WORD MSG12A
 .WORD MSG12B
 .WORD 4600. ; (50-4)*2
 .WORD 5200. ; (50+2)*2

PAT.PT: .WORD PAT0 ; TABLE OF POINTERS WHICH POINT TO THE
 .WORD PAT1 ; PATTERNS USED BY THE DATA TEST
 .WORD PAT2

-35

2187	002442	002634	.WORD	PAT3
2188	002444	002674	.WORD	PAT4
2189	002446	002734	.WORD	PAT5
2190	002450	002774	.WORD	PAT6
2191	002452	003034	.WORD	PAT7
2192	002454	003074	.WORD	PAT8
2193	002456	003134	.WORD	PAT9
2194	002460	003174	.WORD	PAT10
2195	002462	003234	.WORD	PAT11
2196	002464	003274	.WORD	PAT12
2197	002466	003334	.WORD	PAT13
2198	002470	003374	.WORD	PAT14
2199	002472	003434	.WORD	PAT15

; PATTERNS 0 THRU 15

2203	002474	165555	PAT0: .WORD	165555	; PATTERN 0
2204	002476	133333	.WORD	133333	
2205	002500	165555	.WORD	165555	
2206	002502	133333	.WORD	133333	
2207	002504	165555	.WORD	165555	
2208	002506	133333	.WORD	133333	
2209	002510	165555	.WORD	165555	
2210	002512	133333	.WORD	133333	
2211	002514	165555	.WORD	165555	
2212	002516	133333	.WORD	133333	
2213	002520	165555	.WORD	165555	
2214	002522	133333	.WORD	133333	
2215	002524	165555	.WORD	165555	
2216	002526	133333	.WORD	133333	
2217	002530	165555	.WORD	165555	
2218	002532	133333	.WORD	133333	

2220	002534	000001	PAT1: .WORD	000001	; PATTERN 1
2221	002536	000003	.WORD	000003	
2222	002540	000007	.WORD	000007	
2223	002542	000017	.WORD	000017	
2224	002544	000037	.WORD	000037	
2225	002546	000077	.WORD	000077	
2226	002550	000177	.WORD	000177	
2227	002552	000377	.WORD	000377	
2228	002554	000777	.WORD	000777	
2229	002556	001777	.WORD	001777	
2230	002560	003777	.WORD	003777	
2231	002562	007777	.WORD	007777	
2232	002564	017777	.WORD	017777	
2233	002566	037777	.WORD	037777	
2234	002570	077777	.WORD	077777	
2235	002572	177777	.WORD	177777	

2237	002574	177776	PAT2: .WORD	177776	; PATTERN 2
2238	002576	177774	.WORD	177774	
2239	002600	177770	.WORD	177770	
2240	002602	177760	.WORD	177760	
2241	002604	177740	.WORD	177740	
2242	002606	177700	.WORD	177700	

2243	002610	177600	.WORD	177600	
2244	002612	177400	.WORD	177400	
2245	002614	177000	.WORD	177000	
2246	002616	176000	.WORD	176000	
2247	002620	174000	.WORD	174000	
2248	002622	170000	.WORD	170000	
2249	002624	160000	.WORD	160000	
2250	002626	140000	.WORD	140000	
2251	002630	100000	.WORD	100000	
2252	002632	000000	.WORD	000000	
2253					
2254	002634	000000	PAT3: .WORD	000000	;PATTERN 3
2255	002636	000000	.WORD	000000	
2256	002640	000000	.WORD	000000	
2257	002642	177777	.WORD	177777	
2258	002644	177777	.WORD	177777	
2259	002646	177777	.WORD	177777	
2260	002650	000000	.WORD	000000	
2261	002652	000000	.WORD	000000	
2262	002654	177777	.WORD	177777	
2263	002656	177777	.WORD	177777	
2264	002660	000000	.WORD	000000	
2265	002662	177777	.WORD	177777	
2266	002664	000000	.WORD	000000	
2267	002666	177777	.WORD	177777	
2268	002670	000000	.WORD	000000	
2269	002672	177777	.WORD	177777	
2270					
2271	002674	000000	PAT4: .WORD	000000	;PATTERN 4
2272	002676	010421	.WORD	010421	
2273	002700	021042	.WORD	021042	
2274	002702	031463	.WORD	031463	
2275	002704	042104	.WORD	042104	
2276	002706	052525	.WORD	052525	
2277	002710	063146	.WORD	063146	
2278	002712	073567	.WORD	073567	
2279	002714	104210	.WORD	104210	
2280	002716	114631	.WORD	114631	
2281	002720	125252	.WORD	125252	
2282	002722	135673	.WORD	135673	
2283	002724	146314	.WORD	146314	
2284	002726	156735	.WORD	156735	
2285	002730	167356	.WORD	167356	
2286	002732	177777	.WORD	177777	
2287					
2288	002734	052525	PAT5: .WORD	052525	;PATTERN 5
2289	002736	052525	.WORD	052525	
2290	002740	052525	.WORD	052525	
2291	002742	125252	.WORD	125252	
2292	002744	125252	.WORD	125252	
2293	002746	125252	.WORD	125252	
2294	002750	052525	.WORD	052525	
2295	002752	052525	.WORD	052525	
2296	002754	125252	.WORD	125252	
2297	002756	125252	.WORD	125252	
2298	002760	052525	.WORD	052525	

2299	002762	125252	.WORD	125252
2300	002764	052525	.WORD	052525
2301	002766	125252	.WORD	125252
2302	002770	052525	.WORD	052525
2303	002772	125252	.WORD	125252
2304				
2305	002774	007417	.WORD	007417
2306	002776	007417	.WORD	007417
2307	003000	007417	.WORD	007417
2308	003002	170360	.WORD	170360
2309	003004	170360	.WORD	170360
2310	003006	170360	.WORD	170360
2311	003010	007417	.WORD	007417
2312	003012	007417	.WORD	007417
2313	003014	170360	.WORD	170360
2314	003016	170360	.WORD	170360
2315	003020	007417	.WORD	007417
2316	003022	170360	.WORD	170360
2317	003024	007417	.WORD	007417
2318	003026	170360	.WORD	170360
2319	003030	007417	.WORD	007417
2320	003032	170360	.WORD	170360
2321				
2322	003034	026455	.WORD	026455
2323	003036	026455	.WORD	026455
2324	003040	026455	.WORD	026455
2325	003042	151322	.WORD	151322
2326	003044	151322	.WORD	151322
2327	003046	151322	.WORD	151322
2328	003050	026455	.WORD	026455
2329	003052	026455	.WORD	026455
2330	003054	151322	.WORD	151322
2331	003056	151322	.WORD	151322
2332	003060	026455	.WORD	026455
2333	003062	151322	.WORD	151322
2334	003064	026455	.WORD	026455
2335	003066	151322	.WORD	151322
2336	003070	026455	.WORD	026455
2337	003072	151322	.WORD	151322
2338				
2339	003074	165555	.WORD	165555
2340	003076	133333	.WORD	133333
2341	003100	165555	.WORD	165555
2342	003102	133333	.WORD	133333
2343	003104	165555	.WORD	165555
2344	003106	133333	.WORD	133333
2345	003110	165555	.WORD	165555
2346	003112	133333	.WORD	133333
2347	003114	165555	.WORD	165555
2348	003116	133333	.WORD	133333
2349	003120	165555	.WORD	165555
2350	003122	133333	.WORD	133333
2351	003124	165555	.WORD	165555
2352	003126	133333	.WORD	133333
2353	003130	165555	.WORD	165555
2354	003132	133333	.WORD	133333

PAT6: ;PATTERN 6

PAT7: ;PATTERN 7

PAT8: ;PATTERN 8

2355					
2356	003134	000001	PAT9:	.WORD	000001 ;PATTERN 9
2357	003136	000002		.WORD	000002
2358	003140	000004		.WORD	000004
2359	003142	000010		.WORD	000010
2360	003144	000020		.WORD	000020
2361	003146	000040		.WORD	000040
2362	003150	000100		.WORD	000100
2363	003152	000200		.WORD	000200
2364	003154	000400		.WORD	000400
2365	003156	001000		.WORD	001000
2366	003160	002000		.WORD	002000
2367	003162	004000		.WORD	004000
2368	003164	010000		.WORD	010000
2369	003166	020000		.WORD	020000
2370	003170	040000		.WORD	040000
2371	003172	100000		.WORD	100000
2372					
2373	003174	177776	PAT10:	.WORD	177776 ;PATTERN 10
2374	003176	177775		.WORD	177775
2375	003200	177773		.WORD	177773
2376	003202	177767		.WORD	177767
2377	003204	177757		.WORD	177757
2378	003206	177737		.WORD	177737
2379	003210	177677		.WORD	177677
2380	003212	177577		.WORD	177577
2381	003214	177377		.WORD	177377
2382	003216	176777		.WORD	176777
2383	003220	175777		.WORD	175777
2384	003222	173777		.WORD	173777
2385	003224	167777		.WORD	167777
2386	003226	157777		.WORD	157777
2387	003230	137777		.WORD	137777
2388	003232	077777		.WORD	077777
2389					
2390	003234	172666	PAT11:	.WORD	172666 ;PATTERN 11
2391	003236	155555		.WORD	155555
2392	003240	172666		.WORD	172666
2393	003242	155555		.WORD	155555
2394	003244	172666		.WORD	172666
2395	003246	155555		.WORD	155555
2396	003250	172666		.WORD	172666
2397	003252	155555		.WORD	155555
2398	003254	172666		.WORD	172666
2399	003256	155555		.WORD	155555
2400	003260	172666		.WORD	172666
2401	003262	155555		.WORD	155555
2402	003264	172666		.WORD	172666
2403	003266	155555		.WORD	155555
2404	003270	172666		.WORD	172666
2405	003272	155555		.WORD	155555
2406					
2407	003274	077777	PAT12:	.WORD	077777 ;PATTERN 12
2408	003276	137777		.WORD	137777
2409	003300	157777		.WORD	157777
2410	003302	167777		.WORD	167777

2411	003304	173777	.WORD	173777
2412	003306	175777	.WORD	175777
2413	003310	176777	.WORD	176777
2414	003312	177377	.WORD	177377
2415	003314	177577	.WORD	177577
2416	003316	177677	.WORD	177677
2417	003320	177737	.WORD	177737
2418	003322	177757	.WORD	177757
2419	003324	177767	.WORD	177767
2420	003326	177773	.WORD	177773
2421	003330	177775	.WORD	177775
2422	003332	177776	.WORD	177776
2423				
2424	003334	153333	PAT13: .WORD	153333
2425	003336	066667	.WORD	066667
2426	003340	153333	.WORD	153333
2427	003342	066667	.WORD	066667
2428	003344	153333	.WORD	153333
2429	003346	066667	.WORD	066667
2430	003350	153333	.WORD	153333
2431	003352	066667	.WORD	066667
2432	003354	153333	.WORD	153333
2433	003356	066667	.WORD	066667
2434	003360	153333	.WORD	153333
2435	003362	066667	.WORD	066667
2436	003364	153333	.WORD	153333
2437	003366	066667	.WORD	066667
2438	003370	153333	.WORD	153333
2439	003372	066667	.WORD	066667
2440				
2441	003374	000000	PAT14: .WORD	000000
2442	003376	177777	.WORD	177777
2443	003400	177777	.WORD	177777
2444	003402	177777	.WORD	177777
2445	003404	177777	.WORD	177777
2446	003406	177777	.WORD	177777
2447	003410	177777	.WORD	177777
2448	003412	177777	.WORD	177777
2449	003414	177777	.WORD	177777
2450	003416	177777	.WORD	177777
2451	003420	177777	.WORD	177777
2452	003422	177777	.WORD	177777
2453	003424	177777	.WORD	177777
2454	003426	177777	.WORD	177777
2455	003430	177777	.WORD	177777
2456	003432	177777	.WORD	177777
2457				
2458	003434	177777	PAT15: .WORD	177777
2459	003436	000000	.WORD	000000
2460	003440	000000	.WORD	000000
2461	003442	000000	.WORD	000000
2462	003444	000000	.WORD	000000
2463	003446	000000	.WORD	000000
2464	003450	000000	.WORD	000000
2465	003452	000000	.WORD	000000
2466	003454	000000	.WORD	000000

;PATTERN 13

;PATTERN 14

;PATTERN 15

2467	003456	000000	.WORD	000000
2468	003460	000000	.WORD	000000
2469	003462	000000	.WORD	000000
2470	003464	000000	.WORD	000000
2471	003466	000000	.WORD	000000
2472	003470	000000	.WORD	000000
2473	003472	000000	.WORD	000000

;DPB (DATA PARAMETER BLOCK)

2477	003474	000	DPB.A: .BYTE	0
2478	003475	000	.BYTE	0
2479	003476	000	.BYTE	0
2480	003477	000	.BYTE	0
2481	003500	000000	.WORD	0
2482	003502	044134	.WORD	BUFFER
2484	003504	000	.BYTE	0
2486	003505	000	.BYTE	0
2488	003506	000000	.WORD	0
2489	003510	003574	.WORD	RP.REG

```

;(0) DRIVE NUMBER
;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
;(2) COMMAND
;(3) PSEL AND A17 AND A16
;(4) WORD COUNT (MUST BE NEG.)
;(6) BUFFER ADDRESS OR
REGISTER TABLE POINTER
;(10) SECTOR ADDRESS OR
FIRST REG. INDEX
;(11) TRACK ADDRESS OR
LAST REG. INDEX
;(12) CYLINDER ADDRESS
;(14) ERROR TABLE POINTER
POINTS TO THE FIRST OF TWENTY
LOCATIONS OF WHERE THE DRIVER
IS TO STORE THE RH11/RP04
REGISTERS ON AN ERROR. IF LEFT
ZERO REGISTERS ARE NOT SAVED.
;(16) STATUS/ERROR INDICATOR
BIT15=1=>ERROR OCCURRED
BIT07=1=>DONE
BIT14-BIT09 AND BIT06-BIT03
INDICATE TYPE OF ERROR

```

2495	003512	000000	.WORD	0
2501	003514	000	DPB.B: .BYTE	0
2502	003515	000	.BYTE	0
2503	003516	000	.BYTE	0
2504	003517	000	.BYTE	0
2505	003520	177776	.WORD	-2
2506	003522	044134	.WORD	BUFFER
2508	003524	000	.BYTE	0
2510	003525	000	.BYTE	0
2512	003526	000000	.WORD	0
2513	003530	003574	.WORD	RP.REG

```

;(0) DRIVE NUMBER
;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
;(2) COMMAND
;(3) PSEL AND A17 AND A16
;(4) WORD COUNT (MUST BE NEG.)
;(6) BUFFER ADDRESS OR
REGISTER TABLE POINTER
;(10) SECTOR ADDRESS OR
FIRST REG. INDEX
;(11) TRACK ADDRESS OR
LAST REG. INDEX
;(12) CYLINDER ADDRESS
;(14) ERROR TABLE POINTER
POINTS TO THE FIRST OF TWENTY
LOCATIONS OF WHERE THE DRIVER
IS TO STORE THE RH11/RP04
REGISTERS ON AN ERROR. IF LEFT
ZERO REGISTERS ARE NOT SAVED.
;(16) STATUS/ERROR INDICATOR
BIT15=1=>ERROR OCCURRED
BIT07=1=>DONE
BIT14-BIT09 AND BIT06-BIT03
INDICATE TYPE OF ERROR

```

2519	003532	000000	.WORD	0
------	--------	--------	-------	---

```

2523
2524
2525 003534 000 DPB.C: .BYTE 0
2526 003535 000 .BYTE 0
2527 003536 000 .BYTE 0
2528 003537 000 .BYTE 0
2529 003540 177776 .WORD -2
2530 003542 044134 .WORD BUFFER
2531
2532 003544 000 .BYTE 0
2533
2534 003545 000 .BYTE 0
2535
2536 003546 000000 .WORD 0
2537 003550 003574 .WORD RP.REG
2538
2539
2540
2541
2542
2543 003552 000000 .WORD 0
2544
2545
2546
2547
2548
2549 003554 000 DTADPB: .BYTE 0
2550 003555 000 .BYTE 0
2551 003556 000 .BYTE 0
2552 003557 000 .BYTE 0
2553 003560 000000 .WORD 0
2554 003562 044134 .WORD BUFFER
2555
2556 003564 000 .BYTE 0
2557
2558 003565 000 .BYTE 0
2559
2560 003566 000000 .WORD 0
2561 003570 003574 .WORD RP.REG
2562
2563
2564
2565
2566
2567 003572 000000 .WORD 0
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577 003574 000000 RP.REG: .WORD 0
2578 003576 000000 .WORD 0

```

```

;INDICATE TYPE OF ERROR
;(0) DRIVE NUMBER
;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
;(2) COMMAND
;(3) PSEL AND A17 AND A16
;(4) WORD COUNT (MUST BE NEG.)
;(6) BUFFER ADDRESS OR
REGISTER TABLE POINTER
;(10) SECTOR ADDRESS OR
FIRST REG. INDEX
;(11) TRACK ADDRESS OR
LAST REG. INDEX
;(12) CYLINDER ADDRESS
;(14) ERROR TABLE POINTER
POINTS TO THE FIRST OF TWENTY
LOCATIONS OF WHERE THE DRIVER
IS TO STORE THE RH11/RP04
REGISTERS ON AN ERROR. IF LEFT
ZERO REGISTERS ARE NOT SAVED.
;(16) STATUS/ERROR INDICATOR
BIT15=1=>ERROR OCCURRED
BIT07=1=>DONE
BIT14-BIT09 AND BIT06-BIT03
;INDICATE TYPE OF ERROR
;(0) DRIVE NUMBER
;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
;(2) COMMAND
;(3) PSEL AND A17 AND A16
;(4) WORD COUNT (MUST BE NEG.)
;(6) BUFFER ADDRESS OR
REGISTER TABLE POINTER
;(10) SECTOR ADDRESS OR
FIRST REG. INDEX
;(11) TRACK ADDRESS OR
LAST REG. INDEX
;(12) CYLINDER ADDRESS
;(14) ERROR TABLE POINTER
POINTS TO THE FIRST OF TWENTY
LOCATIONS OF WHERE THE DRIVER
IS TO STORE THE RH11/RP04
REGISTERS ON AN ERROR. IF LEFT
ZERO REGISTERS ARE NOT SAVED.
;(16) STATUS/ERROR INDICATOR
BIT15=1=>ERROR OCCURRED
BIT07=1=>DONE
BIT14-BIT09 AND BIT06-BIT03
;INDICATE TYPE OF ERROR

;SAVE RH11/RP04 REGISTERS HERE ON ERROR
;RHCS1 (776700) CONTROL & STATUS #1
;RHWC (776702) WORD COUNT

```


2579	003600	000000	.WORD	0	;RHBA (776704) BUS ADDRESS
2580	003602	000000	.WORD	0	;RHDA (776706) DESIRED SECTOR/TRACK
2581	003604	000000	.WORD	0	;RHCS2 (776710) CONTROL & STATUS #2
2582	003606	000000	.WORD	0	;RHDS1 (776712) DISK STATUS
2583	003610	000000	.WORD	0	;RHER1 (776714) ERROR REG. #1
2584	003612	000000	.WORD	0	;RHAS (776716) ATTENTION SUMMARY
2585	003614	000000	.WORD	0	;RHLA (776720) LOOK AHEAD
2586	003616	000000	.WORD	0	;RHDB (776722) DATA BUFFER
2587	003620	000000	.WORD	0	;RHMR (776724) MAINTAINABILITY
2588	003622	000000	.WORD	0	;RHDT (776726) DRIVE TYPE
2589	003624	000000	.WORD	0	;RHSN (776730) SERIAL NUMBER
2590	003626	000000	.WORD	0	;RHOF (776732) OFFSET
2591	003630	000000	.WORD	0	;RHCA (776734) DESIRED CYLINDER
2592	003632	000000	.WORD	0	;RHCC (776736) CURRENT CYLINDER
2593	003634	000000	.WORD	0	;RHER2 (776740) ERROR REG #2
2594	003636	000000	.WORD	0	;RHER3 (776742) ERROR REG #3
2595	003640	000000	.WORD	0	;RHEC1 (776744) ECC POSITION
2596	003642	000000	.WORD	0	;RHEC2 (776746) ECC PATTERN

2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652

003644

003644 040615
003646 042124
003650 043364
003652 043770

003654 040663
003656 042143
003660 043370
003662 043774

003664 040723
003666 042241
003670 043410
003672 044000

.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

\$ERRTB:

;*EM AND DH ARE ASCIZ MESSAGES, DT IS A STRING OF WORDS THAT POINT TO THE
;*DATA TO BE TYPED AND DF IS A STRING OF DATA THAT TELL HOW THE DT WORDS
;*ARE TO BE TYPED. IF ANY OF THE POINTERS ARE NOT NEEDED FOR A PARTICULAR
;*ERROR IT IS REPLACED WITH A ZERO.
;*EACH OF THE ITEMS BELOW REFER TO THE ERROR NUMBER AND INDICATE
;*THE INFORMATION THAT WILL BE TYPED WHEN THE ERROR OCCURS.
;*UNLESS STATED OTHER ALL NUMBERS ARE OCTAL

;* ERROR ITEM 1
;* ILLEGAL RH11(SC=0)INTERRUPT OCCURRED
;* TEST ERR PC
;* \$TMPO \$ERRPC

EM1
DH1
DT1
DF1

;* ERROR ITEM 2
;* ILLEGAL RPO4 INTERRUPT OCCURRED
;* TEST ERR PC DRIVE RHAS RHDS RHER1 RHER2 RHER3
;* \$TMPO \$ERRPC \$REG1 \$REG5 RPERRS RPERRS+2 RPERRS+4 RPERRS+6

EM2
DH2
DT2
DF2

;* ERROR ITEM 3
;* MASSBUS PARITY ERROR (MCPE=1)
;* TEST ERR PC ADDRESS DATA
;* \$TMPO \$ERRPC RD.ADR RD.WRD

EM3
DH3
DT3
DF3

;* ERROR ITEM 4


```

2653      : * MASSBUS PARITY ERROR (PAR=1)
2654      : * TEST   ERR PC ADDRESS GDDATA BDDATA
2655      : * STMPD  $ERRPC WRT.ADR WRT.WD  RD.WRD
2656
2657 003674 040760      EM4
2658 003676 042276      DH4
2659 003700 043420      DT4
2660 003702 044004      DF4
2661
2662      : * ERROR ITEM 5
2663      : * OFFLINE, NONEXIS. ENT, OR UNSAFE DRIVE INTERRUPTED
2664      : * TEST   ERR PC DRIVE  RHAS  RHDS  RHER1  RHER2  RHER3
2665      : * STMPD  $ERRPC $REG1  $REG5  RPERRS  RPERRS+2  RPERRS+4  RPERRS+6
2666
2667 003704 041014      EM5
2668 003706 042143      DH2
2669 003710 043370      DT2
2670 003712 043774      DF2
2671
2672      ; * ERROR ITEM 6 -- NOT USED
2673
2674 003714 000000      0
2675 003716 000000      0
2676 003720 000000      0
2677 003722 000000      0
2678
2679      ; * ERROR ITEM 7 -- NOT USED
2680
2681 003724 000000      0
2682 003726 000000      0
2683 003730 000000      0
2684 003732 000000      0
2685
2686      : * ERROR ITEM 10
2687      : * RH11/RP04 FAILED TO RESPOND TO ADDRESSING
2688      : * RHCS1  ERR PC
2689      : * RH.ADR  $ERRPC
2690
2691 003734 041077      EM10
2692 003736 042345      DH10
2693 003740 043452      DT10
2694 003742 044010      DF10
2695
2696      : * ERROR ITEM 11
2697      : * DRIVE SELECTED IS NOT ONLINE
2698      : * DRIVE  ERR PC
2699      : * $REG2  $ERRPC
2700
2701 003744 041151      EM11
2702 003746 042364      DH11
2703 003750 043456      DT11
2704 003752 044014      DF11
2705
2706      : * ERROR ITEM 12
2707      : * IMPROPER HEADER DATA
2708      : * TEST   ERR PC TST PC DRIVE  CYLNDR TRACK  SECTOR

```

```

2709      ;*      STMPO  $ERRPC $REGO  CHKDRV  CYL.DS  TRK.DS  SEC.DS
2710      ;*      GDCYL  GDTRK  GDSCTR  BDCYL  BDTRK  BDSCTR
2711      ;*      CYL.DS  TRK.DS  SEC.DS  CYL.RD  TRK.RD  SEC.RD
2712      ;*      CYLNDR, TRACK, AND SECTOR ARE DECIMAL
2713
2714      003754  041206      EM12
2715      003756  042403      DH12
2716      003760  043462      DT12
2717      003762  044020      DF12
2718
2719      ;*      ERROR ITEM 13
2720      ;*      DATA COMPARE FAILURE
2721      ;*      TEST   ERR PC  TST PC  DRIVE  CYLNDR  TRACK  SECTOR
2722      ;*      STMPO  $ERRPC $REGO  CHKDRV  CYL.DS  TRK.DS  SEC.DS
2723      ;*      GDDAT  BDDAT  WRDCNT  GDADR  BDADR
2724      ;*      $GDDAT $BDDAT $REG4  $GDADR $BDADR
2725      ;*      CYLNDR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
2726
2727      003764  041233      EM13
2728      003766  042403      DH12
2729      003770  043514      DT13
2730      003772  044030      DF13
2731
2732      ;*      ERROR ITEM 14 -- FOLLOWS #13
2733      ;*      $GDDAT $BDDAT $REG4  $GDADR $BDADR
2734
2735      003774  000000      0
2736      003776  000000      0
2737      004000  043532      DT13A
2738      004002  044040      DF14
2739
2740      ;*      ERROR ITEM 15
2741      ;*      DATA COMPARE FAILURE
2742      ;*      TEST   ERR PC  TST PC  DRIVE  CYLNDR  TRACK  SECTOR
2743      ;*      STMPO  $ERRPC $REGO  CHKDRV  CYL.DS  TRK.DS  SEC.DS
2744      ;*      GDDAT  BDDAT  WRDCNT  GDADR  BDADR
2745      ;*      $GDDAT $BDDAT $REG4  $GDADR $BDADR
2746      ;*      CYLNDR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
2747
2748      004004  041233      EM13
2749      004006  042403      DH12
2750      004010  043514      DT13
2751      004012  044030      DF13
2752
2753      ;*      ERROR ITEM 16 -- FOLLOWS #15
2754      ;*      $GDDAT $BDDAT $REG4  $GDADR $BDADR
2755
2756      004014  000000      0
2757      004016  000000      0
2758      004020  043532      DT13A
2759      004022  044040      DF14
2760
2761      ;*      ERROR ITEM 17
2762      ;*      DISK ERROR IN TIMING TEST
2763      ;*      TEST   ERR PC  DRIVE  RHCS1  RHDS  RHER1  RHER2  RHER3
2764      ;*      STMPO  $ERRPC  CHKDRV  RP.REG  RP.REG+12  RP.REG+14  RP.REG+40  RP.REG+42
  
```


2765
2766 004024 041260
2767 004026 042617
2768 004030 043544
2769 004032 044044

EM17
DH17
DT17
DF17

2770
2771 : * ERROR ITEM 20
2772 : * CLOCK (KW11-P) OVERFLOW IN TIMING TEST
2773 : * TEST ERR PC DRIVE RHCS1 RHDS RHER1 RHER2 RHER3
2774 : * STMPO \$ERRPC CHKDRV RP.REG RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42

2775
2776 004034 041312
2777 004036 042617
2778 004040 043544
2779 004042 044044

EM20
DH17
DT17
DF17

2780
2781 : * ERROR ITEM 21
2782 : * DATA COMPARE FAILURE
2783 : * TEST ERR PC TST PC DRIVE CYLNR TRACK
2784 : * STMPO \$ERRPC \$REGD CHKDRV CYL.DS TRK.DS
2785 : * GDDAT BDDAT WRDCNT SECTOR
2786 : * \$REG1 \$BDDAT \$REG4 \$REG1
2787 : * CYLINDR, TRACK, WRDCNT, AND SECTOR ARE DECIMAL

2788
2789 004044 041233
2790 004046 042715
2791 004050 043564
2792 004052 044050

EM13
DH21
DT21
DF21

2793
2794 : * ERROR ITEM 22--FOLLOWS #21
2795 : * \$REG1 \$BDDAT \$REG4 \$REG1

2796
2797 004054 000000
2798 004056 000000
2799 004060 043600
2800 004062 044060

0
0
DT21A
DF22

2801
2802 : * *****
2803 : * *****
2804 : * ERROR ITEMS 23-40 NOT USED
2805 : * ERROR ITEMS 41-46 WILL HAVE AN EM THAT
2806 : * VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM:
2807 : * RH11/RP04 ERROR (MESSAGE)
2808 : * WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING:
2809 : * 1) OFFLINE OR UNSAFE DRIVE REQUESTED
2810 : * 2) UNLOADED DRIVE REQUESTED
2811 : * 3) PERSISTENT UNSAFE
2812 : * 4) PARITY ERROR OCCURRED
2813 : * 5) FATAL PARITY ERROR
2814 : * 6) SOFTWARE TIMEOUT ON THIS DRIVE
2815 : * 7) SOFTWARE TIMEOUT ON ANOTHER DRIVE
2816 : * 8) ERROR OCCURRED DURING I/O OPERATION
2817 : * 9) ERROR OCCURRED DURING NON-I/O OPERATION
2818 : * 10) UNSAFE OCCURRED
2819 : * 11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
2820


```

2877      : *      RP.REG+14  RP.REG+40  RP.REG+42  RP.REG+2  RP.REG+4  RP.REG+22
2878      : *      CYLNR,TRACK, AND SECTOR ARE DECIMAL
2879
2880 004124 041361      EM41
2881 004126 042403      DH12
2882 004130 043722      DT45
2883 004132 044120      DF45
2884
2885      : *      ERROR ITEM 46
2886      : *      FATAL WRITE CHECK ERROR (MESSAGE)
2887      : *      TEST  ERR PC  TST PC  DRIVE  CYLNR  TRACK  SECTOR
2888      : *      $TMPD $ERRPC $REGO  CHKDRV  CYL.DS  TRK.DS  SEC.DS
2889      : *      RHCS1  RHCS2  RHDS  RHCC  RHCA  RHDA
2890      : *      RP.REG  RP.REG+10  RP.REG+12  RP.REG+36  RP.REG+34  RP.REG+06
2891      : *      RHER1  RHER2  RHER3  RHC  RHBA  RHDB
2892      : *      RP.REG+14  RP.REG+40  RP.REG+42  RP.REG+2  RP.REG+4  RP.REG+22
2893      : *      CYLNR,TRACK, AND SECTOR ARE DECIMAL
2894
2895 004134 041401      EM46
2896 004136 042403      DH12
2897 004140 043722      DT45
2898 004142 044120      DF45
2899
2900      000400      A16=400
2901      001000      A17=1000
2902      000114      PARE70=114

```

```

2903
2904           .SBTTL  START OF PROGRAM
2905
2906
2907 004144 012737 177777 001252 START3: MOV    #-1, @#BUSADR ;GET BUSADR FLAG
2908 004152 000402                BR      STRT1A
2909 004154 005037 001252          START1: CLR    @#BUSADR ;CLR BUSADR FLAG
2910 004160 012737 137777 001260 STRT1A: MOV    #137777, TSTNMS ;SELECT TESTS 0-15 AND 17
2911 004166 005037 001250          CLR    @#CNTRLC ;NO CONTROL "C"
2912 004172 012700 001650          MOV    #DFLT, R0 ;DEFAULT PARAMETERS POINTER
2913 004176 012701 002116          MOV    #PRMO, R1 ;TABLE POINTER
2914 004202 010102                MOV    R1, R2 ;STOP ADDRESS
2915 004204 012021                1$: MOV    (R0)+, (R1)+ ;MOVE DEFAULT PARAMETERS INTO
2916 004206 020002                CMP    R0, R2 ;RUN TIME TABLES ** DONE?
2917 004210 103775                BLO   1$ ;NO--BRANCH
2918 004212 012700 003074          MOV    #PAT8, R0 ;PATO DEFAULTS TO PATTERN 8
2919 004216 012701 002474          MOV    #PATO, R1
2920 004222 012021                2$: MOV    (R0)+, (R1)+
2921 004224 020027 003134          CMP    R0, #PAT9
2922 004230 103774                BLO   2$
2923 004232 000411                BR      START
2924 004234 012737 177777 001252 START4: MOV    #-1, @#BUSADR ;SET BUSADR FLAG
2925 004242 000402                BR      STRT2A
2926 004244 005037 001252          START2: CLR    @#BUSADR ;CLR BUSADR FLAG
2927 004250 012737 177777 001250 STRT2A: MOV    #-1, @#CNTRLC ;SET CONTROL "C" FLAG
2928 004256 000005          START: RESET
2929 004260 012737 000340 177776  MOV    #340, @#PS ;; LOCK OUT ALL INTERRUPTS
2930 004266 012706 001100          MOV    #SCMTAG, R6 ;; FIRST LOCATION TO BE CLEARED
2931 004272 005026                CLR    (R6)+ ;; CLEAR MEMORY LOCATION
2932 004274 022706 001136          CMP    #STKS, R6 ;; DONE?
2933 004300 001374                BNE   -6 ;; LOOP BACK IF NO
2934 004302 012706 001100          MOV    #STACK, SP ;; SETUP THE STACK POINTER
2935 004306 012737 020470 000020  MOV    #SCOPE, @#IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
2936 004314 012737 000340 000022  MOV    #340, @#IOTVEC+2 ;; LEVEL 7
2937 004322 012737 015230 000030  MOV    #ERROR, @#EMTVEC ;; EMT VECTOR FOR ERROR ROUTINE
2938 004330 012737 000340 000032  MOV    #340, @#EMTVEC+2 ;; LEVEL 7
2939 004336 012737 021004 000034  MOV    #STRAP, @#TRAPVEC ;; TRAP VECTOR FOR TRAP CALLS
2940 004344 012737 000340 000036  MOV    #340, @#TRAPVEC+2 ;; LEVEL 7
2941 004352 005037 001176          CLR    #TIMES ;; INITIALIZE NUMBER OF ITERATIONS
2942 004356 005037 001200          CLR    #ESCAPE ;; CLEAR THE ESCAPE ON ERROR ADDRESS
2943 004362 112737 000001 001115  MOVB  #1, #SERMAX ;; ALLOW ONE ERROR PER TEST
2944 004370 012737 004370 001106  MOV    #., #SLPADR ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
2945 004376 012737 004376 001110  MOV    #., #SLPERR ;; SETUP THE ERROR LOOP ADDRESS
2946 004404 012700 001152          MOV    #REGAD, R0 ;; FIRST ADDRESS
2947 004410 005020                1$: CLR    (R0)+ ;; CLEAR VARIABLE STORAGE
2948 004412 022700 001202          CMP    #SBELL, R0 ;; DONE?
2949 004416 001374                BNE   1$ ;; NO--BRANCH
2950 004420 013737 001410 001144  MOV    @#TPS, @#STPB ;; SETUP THE STATUS AND BUFFER REG'S
2951 004426 013737 001412 001144  MOV    @#TPB, @#STPB ;; FOR THE TYPE ROUTINE
2952 004434 063727 001250 000000  ADD    @#CNTRLC, #0 ;; TYPE "ID" IF START AT 200 OR
2953 004442 103433                BCS   SRTINT ;; THE 1ST START AT 204
2954 004444 104400 004452          TYPE  #+4 ;; TYPE ASCIZ STRING
2955 004450 000430                BR     64$ ;; GET OVER THE ASCIZ
2956                ;; .ASCIZ <15><12>/MD-11-DERPK-C "MECHANICAL & READ-WRITE" TEST/
2957 004532                64$:
2958 004532 004737 022340          SRTINT: JSR   PC, @#LP.AVL ;CHECK FOR A LINE PRINTER

```



```

3015 005024 005200          35:  INC      RO
3016 005026 106301          ASLB    R1
3017 005030 001367          BNE     25
3018 005032 005037 032322  45:  CLR      @#SEEKFG      ;CLEAR SEEK FLAG
3019 005036 032737 000400 001246  BIT     @#SW08,@#C.SW  ;DO SEEK BEFORE DATA TRANSFER?
3020 005044 001002          BNE     55             ;YES--BRANCH
3021 005046 005137 032322  COM     @#SEEKFG      ;NO
3022 005052
3023 005052 104400 005060  55:  TYPE    .+4           ;;TYPE ASCIZ STRING
3024 005056 000415          BR      66$           ;;GET OVER THE ASCIZ
3025
3026 005112          ;;.ASCIZ <15><12>/DRIVE(S) TO BE TESTED /
3027 005112 005037 015126  66$:  CLR      @#SENDCT     ;DETERMINE PASSES TO MAKE AND
3028 005116 005000          CLR      RO           ;THE DRIVES TO BE TESTED
3029 005120 013701 001256  MOV     @#DRVSEL,R1   ;ANY DRIVES SELECTED?
3030 005124 001010          BNE     65             ;YES--BRANCH
3031 005126 104400 005134  TYPE    .+4           ;;TYPE ASCIZ STRING
3032 005132 000403          BR      67$           ;;GET OVER THE ASCIZ
3033
3034 005142          ;;.ASCIZ /NONE/
3035 005142 000137 014744  67$:  JMP      @#SEOP       ;GO TO END OF PROGRAM
3036 005146 006201          ASR     R1             ;REPORT THE DRIVES TO BE TESTED
3037 005150 103013          BCC     75
3038 005152 005237 015126  INC     @#SENDCT     ;GIVE THIS DRIVE A PASS
3039 005156 010046          MOV     RO,-(SP)     ;SAVE RO FOR TYPEOUT
3040 005160 104404          TYPPOS ;GO TYPE--OCTAL ASCII
3041 005162 001          .BYTE  1             ;TYPE 1 DIGIT(S)
3042 005163 000          .BYTE  0             ;SUPPRESS LEADING ZEROS
3043 005164 005701          TST     R1            ;MORE DRIVES?
3044 005166 001406          BEQ     85            ;NO--BRANCH
3045 005170 104400 005176  TYPE    .+4           ;;TYPE ASCIZ STRING
3046 005174 000401          BR      68$           ;;GET OVER THE ASCIZ
3047
3048 005200          ;;.ASCIZ ",,"
3049 005200 005200  68$:  INC     RO           ;FORM DRIVE NUMBER
3050 005202 000761          BR      65
3051 005204 013737 015126 015120  85:  MOV     @#SENDCT,@#SEOPCT
3052 005212 005737 001262  TST     @#CLKSTA     ;KW11-P AVAILABLE
3053 005216 003041          BGT     RSTART1     ;YES--BRANCH
3054 005220 032737 003600 001260  BIT     @#3600,@#TSTNMS ;NO--ANY TIMING TESTS TO BE PERFORMED?
3055 005226 001435          BEQ     RSTART1     ;NO--BRANCH
3056 005230 104400 005236  TYPE    .+4           ;;TYPE ASCIZ STRING
3057 005234 000432          BR      69$           ;;GET OVER THE ASCIZ
3058
3059 005322          ;;.ASCIZ <15><12>/NO KW11-P CLOCK--TEST 7-12 WILL NOT BE PERFORMED/
3060 005322 005037 001272  69$:  RSTART1: CLR     CHKDRV   ;INIT. THE CHECK DRIVE KEY
3061 005326 012737 000001 001274  MOV     @#1,DRVMSK  ;START TO CHECK DESIRED DRIVES
3062 005334 033737 001274 001256  RSTART2: BIT     DRVMSK,DRVSEL ;IS THIS DRIVE SELECTED?
3063 005342 001006          RSTART3: BNE     DRVOK   ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
3064 005344 005237 001272  RESTART: INC     CHKDRV  ;MOVE TO NEXT DRIVE NUMBER
3065 005350 106337 001274          ASLB    DRVMSK     ;POSITION THE MASK
3066 005354 103762          BCS     RSTART1     ;BRANCH IF THE DRIVE NUMBER NEEDS INITIALIZED
3067 005356 000766          BR      RSTART2
3068
3069 005360 013702 001272  DRVOK: MOV     @#CHKDRV,R2 ;PICKUP THE DRIVE NUMBER
3070 005364 105762 032242  TSTB   DRVSTA(R2)   ;IS DESIRED DRIVE ON-LINE?

```



```

3071 005370 003002          BGT      1$          ;YES, BRANCH
3072 005372 104011          ERROR    11          ;DRIVE SELECTED IS NOT ONLINE
3073 005374 000763          BR       RESTART    ;NO-- RETURN
3074 005376 010237 003474    1$:     MOV     R2, @#DPB.A ;SET THE DRIVE NUMBER INTO THE DPB'S
3075 005402 010237 003514    MOV     R2, @#DPB.B
3076 005406 010237 003534    MOV     R2, @#DPB.C
3077 005412 010237 003554    MOV     R2, @#DTADPB
3078 005416 004737 022660    JSR     PC, @#LDCMD ;LOAD COMMAND INTO DPB.B AND DPB.C
3079 005422 012737 014744 001270    MOV     @#EOP, @#BYPASS ;IF ERROR GO TO END OF PROGRAM
3080 005430 112737 000020 003475    MOV     @#FMT22/1D256, @#DPB.A+1
3081 005436 012737 000143 003476    MOV     @#SETFORM, @#DPB.A+2 ;SETFORM=COMMAND
3082 005444 004037 022724    JSR     RD, @#CALL.A ;GO EXECUTE THE COMMAND
3083 005450 012737 000107 003476    MOV     @#RECAL, @#DPB.A+2 ;RECAL=COMMAND
3084 005456 004037 022724    JSR     RD, @#CALL.A ;GO EXECUTE THE COMMAND
3085 005462 012737 000030 003504    MOV     @#RASN, @#DPB.A+10 ;FIRST REG. INDEX
3086 005470 112737 000030 003505    MOV     @#RHSN, @#DPB.A+11 ;LAST REG. INDEX
3087 005476 012737 000141 003476    MOV     @#GETREG, @#DPB.A+2 ;GETREG=COMMAND
3088 005504 004037 022724    JSR     RD, @#CALL.A ;GO EXECUTE THE COMMAND
3089 005510 104400 005516    TYPE    +4          ;;TYPE ASCIZ STRING
3090 005514 000411          BR       64$          ;;GET OVER THE ASCIZ
3091          ;;.ASCIZ      <15><12><12>/TESTING DRIVE /
3092          64$:
3093 005540          MOV     R2, -(SP)    ;;SAVE R2 FOR TYPEOUT
3094 005542 010246 104404    TYPOS   1           ;;GO TYPE--OCTAL ASCII
3095 005544          .BYTE  1           ;;TYPE 1 DIGIT(S)
3096 005545          .BYTE  0           ;;SUPPRESS LEADING ZEROS
3097 005546 104400 040612    TYPE    ,MSG.SP     ;TYPE SPACES
3098 005552 104400 005560    TYPE    +4          ;;TYPE ASCIZ STRING
3099 005556 000410          BR       65$          ;;GET OVER THE ASCIZ
3100          ;;.ASCIZ      /SERIAL NUMBER /
3101          65$:
3102 005600          MOV     #4, R0      ;FOUR DIGITS TO TYPE
3103 005604 012700 000004    MOV     @#BUFFER, R1 ;SERIAL NUMBER
3104 005610 005002          CLR     R2          ;ZERO
3105 005612 006101 044134    2$:     ROL     R1          ;PUT THE NEXT DIGIT
3106 005614 006102          ROL     R2          ;INTO R2
3107 005616 006101          ROL     R1
3108 005620 006102          ROL     R2
3109 005622 006101          ROL     R1
3110 005624 006102          ROL     R2
3111 005626 006101          ROL     R1
3112 005630 006102          ROL     R2
3113 005632 062702 000060    ADD     #'0, R2     ;MAKE IT ASCII
3114 005636 010227          MOV     R2, (PC)+  ;SAVE IT
3115 005640 000000          3$:     .WORD  0
3116 005642 104400 005640    TYPE    ,3$        ;TYPE
3117 005646 005300          DEC     R0          ;ALL DIGITS TYPED?
3118 005650 003357          BGT     2$         ;NO -- BRANCH
3119 005652 104400 001207    TYPE    ,SCRLF

```


E06

```

3176
3177
3178
3179
3180
3181
3182
3183
3184 005656
3185 005656 033737 001430 001260
3186 005664 001002
3187 005666 000137 006004
3188 005672 013737 002120 001176
3189 005700 012737 005766 001106
3190 005706 012737 005656 001110
3191 005714 012737 000000 001102
3192
3193 005722 013737 001102 177570
3194 005730 012737 000107 003476
3195 005736 113737 002126 003524
3196 005744 113737 002124 003525
3197 005752 013737 002122 003526
3198 005760 012737 006002 001270
3199 005766 012706 001100
3200 005772 004037 022724
3201 005776 004037 023032
3202 006002 000004
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216 006004
3217 006004 033737 001432 001260
3218 006012 001002
3219 006014 000137 006146
3220 006020 013737 002132 001176
3221 006026 012737 006130 001106
3222 006034 012737 006004 001110
3223 006042 012737 000001 001102
3224
3225 006050 013737 001102 177570
3226 006056 113737 002146 003524
3227 006064 113737 002150 003544
3228 006072 113737 002142 003525
3229 006100 113737 002144 003545
3230 006106 013737 002134 003526
3231 006114 013737 002136 003546

```

```

:* THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
:* COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC" AT
:* THE COMPLETION OF BOTH COMMANDS STATUS INDICATIONS ARE
:* CHECKED TO INSURE NO ERRORS OCCURRED.
:* THIS TEST WILL BE REPEATED 10 TIMES.

```

```

:*****
†TST0:

```

```

        BIT      2#BITS+<0*2>,TSTNMS ;DO THIS TEST?
        BNE      64$ ;YES--BRANCH
        JMP      TST1 ;NO--GO TO THE NEXT TEST
64$:    MOV      2#RPT0,$TIMES ;GET THE ITERATION COUNT
        MOV      #TEST0,2#SLPADR
        MOV      #TST0,2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
        MOV      #0,2#$TSTNM ;SET UP TEST NUMBER AND
                                ;CLEAR THE ERROR FLAG ($ERFLG)
        MOV      $TSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
        MOV      #RECAL,2#DPB.A+2 ;RECAL=COMMAND
        MOVB     2#FS0,2#DPB.B+10 ;FS
        MOVB     2#FT0,2#DPB.B+11 ;FT
        MOV      2#LC0,2#DPB.B+12 ;LC
        MOV      #EXIT0,2#BYPASS ;GO TO EXIT0 ON ERROR
TEST0:  MOV      #STACK,SP ;SET UP STACK POINTER
        JSR      RO,2#CALL.A ;GO EXECUTE THE COMMAND
        JSR      RO,2#CALL.B ;GO EXECUTE THE COMMAND
EXIT0:  SCOPE ;LOOP

```

```

:*****
:*TEST 1 SEEK/SEEK TEST

```

```

:* THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
:* CYCLE TO "LC" "LT" "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
:* "FC" "FT" "FS". AT THE COMPLETION OF EACH SEEK, THE PROPER
:* INDICATORS ARE EXAMINED TO INSURE PROPER OPERATION.
:* "LC" WILL DEFAULT TO 128 AND "FC", "FT", "LT", "FS", AND "LS"
:* WILL DEFAULT TO 0
:* THIS TEST WILL BE REPEATED 100 TIMES

```

```

:*****
†TST1:

```

```

        BIT      2#BITS+<1*2>,TSTNMS ;DO THIS TEST?
        BNE      64$ ;YES--BRANCH
        JMP      TST2 ;NO--GO TO THE NEXT TEST
64$:    MOV      2#RPT1,$TIMES ;GET THE ITERATION COUNT
        MOV      #TEST1,2#SLPADR
        MOV      #TST1,2#SLPERR ;SETUP THE ERROR LOOP ADDRESS
        MOV      #1,2#$TSTNM ;SET UP TEST NUMBER AND
                                ;CLEAR THE ERROR FLAG ($ERFLG)
        MOV      $TSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
        MOVB     2#FS1,2#DPB.B+10 ;FS
        MOVB     2#LS1,2#DPB.C+10 ;LS
        MOVB     2#FT1,2#DPB.B+11 ;FT
        MOVB     2#LT1,2#DPB.C+11 ;LT
        MOV      2#FC1,2#DPB.B+12 ;FC
        MOV      2#LC1,2#DPB.C+12 ;LC

```

3232 006122 012737 006144 001270
 3233 006130 012706 001100
 3234 006134 004037 023176
 3235 006140 004037 023032
 3236 006144 000004
 3237
 3238
 3239
 3240
 3241
 3242
 3243
 3244
 3245
 3246
 3247
 3248
 3249
 3250
 3251 006146
 3252 006146 033737 001434 001260
 3253 006154 001002
 3254 006156 000137 006330
 3255 006162 013737 002154 001176
 3256 006170 012737 006242 001106
 3257 006176 012737 006146 001110
 3258 006204 012737 000002 001102
 3259
 3260 006212 013737 001102 177570
 3261 006220 113737 002166 003524
 3262 006226 113737 002164 003525
 3263 006234 012737 006326 001270
 3264 006242 012706 001100
 3265 006246 013737 002156 003526
 3266 006254
 3267 006254 004037 023032
 3268 006260 063737 002162 003526
 3269 006266 023737 002160 003526
 3270 006274 002367
 3271 006276 013737 002160 003526
 3272 0063J4
 3273 006304 004037 023032
 3274 006310 163737 002162 003526
 3275 006316 023737 002156 003526
 3276 006324 003767
 3277 006326 000004

```

TEST1: MOV #EXIT1, @#BYPASS ;GO TO EXIT1 ON ERROR
        MOV #STACK, SP ;SET THE STACK POINTER
        JSR RO, @#CALL.C ;GO EXECUTE THE COMMAND
        JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
EXIT1: SCOPE ;LOOP
  
```

 ;*TEST 2 INCREMENT/SEEK TEST

;* THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
 ;* CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC".
 ;* WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS
 ;* "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING
 ;* AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC"
 ;* UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH
 ;* SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
 ;* INSURE PROPER OPERATION.

 ;*TST2:

```

BIT @#BITS+(<2*2>), TSTNMS ;DO THIS TEST?
BNE 64$ ;YES--BRANCH
JMP TST3 ;NO--GO TO THE NEXT TEST
64$: MOV @#RPT2, $TIMES ;GET THE ITERATION COUNT
      MOV #TEST2, @#$LPADR
      MOV #TST2, @#$LPERR ;SETUP THE ERROR LOOP ADDRESS
      MOV #2, @#$TSTNM ;SET UP TEST NUMBER AND
                          ;CLEAR THE ERROR FLAG (SERFLG)
                          ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
      MOV $TSTNM, @#DISPLAY
      MOV @#FS2, @#DPB.B+10 ;FS
      MOV @#FT2, @#DPB.B+11 ;FT
      MOV #EXIT2, @#BYPASS ;GO TO EXIT2 ON ERROR
TEST2: MOV #STACK, SP ;SET UP THE STACK POINTER
        MOV @#FC2, @#DPB.B+12 ;FC
INCSK: JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
        ADD @#IC2, @#DPB.B+12 ;MOVE TO NEXT CYLINDER
        CMP @#LC2, @#DPB.B+12 ;OUT OF CYLINDERS?
        BGE INCSK ;NO--BRANCH
        MOV @#LC2, @#DPB.B+12
DECSK: JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
        SUB @#IC2, @#DPB.B+12
        CMP @#FC2, @#DPB.B+12
        BLE DECSK
EXIT2: SCOPE ;LOOP
  
```

 ;*TEST 3 STEPPING SEEK TEST

;* THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0, 1, 2, 4,
 ;* 8, 16, 32, 64, 128, AND 256. AT THE COMPLETION OF EACH SEEK
 ;* COMMAND THE PROPER INDICATORS ARE EXPLAINED TO INSURE PROPER
 ;* OPERATION.

3278
 3279
 3280
 3281
 3282
 3283
 3284
 3285
 3286
 3287


```

3288 006330          TST3:
3289 006330 033737 001436 001260 BIT      @#BITS+(3*2),TSTNMS ;DO THIS TEST?
3290 006336 001002          BNE      64$          ;YES--BRANCH
3291 006340 000137 006470          JMP      TST4          ;NO--GO TO THE NEXT TEST
3292 006344 013737 002172 001176 64$:  MOV      @#RPT3,$TIMES ;GET THE ITERATION COUNT
3293 006352 012737 006424 001106      MOV      @#TEST3,@#SLPADR
3294 006360 012737 006330 001110      MOV      @#TST3,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
3295 006366 012737 000003 001102      MOV      #3,@#$TSTNM      ;SET UP TEST NUMBER AND
                               ;CLEAR THE ERROR FLAG ($ERFLG)
3296                                ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3297 006374 013737 001102 177570      MOV      $TSTNM,@#DISPLAY
3298 006402 113737 002204 003524      MOVB     @#FS3,@#DPB.B+10 ;FS
3299 006410 113737 002202 003525      MOVB     @#FT3,@#DPB.B+11 ;FT
3300 006416 012737 006466 001270      MOV      @#EXIT3,@#BYPASS ;GO TO BYPASS ON ERROR
3301 006424 012706 001100          TEST3: MOV      @#STACK,SP      ;SET UP THE STACK
3302 006430 013737 002174 003526      MOV      @#FC3,@#DPB.B+12 ;FC
3303 006436 004037 023032          JSR      RO,@#CALL.B      ;GO EXECUTE THE COMMAND
3304 006442 013701 002200          MOV      IC3,R1           ;CYLINDER 1
3305 006446 010137 003526          1$:  MOV      R1,@#DPB.B+12   ;DESIRED CYLINDER
3306 006452 004037 023032          JSR      RO,@#CALL.B      ;GO EXECUTE THE COMMAND
3307 006456 006301          ASL      R1               ;MOVE TO NEXT CYLINDER
3308 006460 020137 002176          CMP      R1,@#LC3        ;DONE?
3309 006464 003770          BLE     1$               ;NO--LOOP
3310 006466 000004          EXIT3: SCOPE

```

```

*****
*TEST 4          OSCILLATING SEEK TEST

```

```

* THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
* TO "FC". "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
* "LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC". AT THE
* COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
* EXAMINED TO INSURE PROPER OPERATION.

```

```

3321
3322 006470          TST4:
3323 006470 033737 001440 001260 BIT      @#BITS+(4*2),TSTNMS ;DO THIS TEST?
3324 006476 001002          BNE      64$          ;YES--BRANCH
3325 006500 000137 007022          JMP      TST5          ;NO--GO TO THE NEXT TEST
3326 006504 013737 002210 001176 64$:  MOV      @#RPT4,$TIMES ;GET THE ITERATION COUNT
3327 006512 012737 006600 001106      MOV      @#TEST4,@#SLPADR
3328 006520 012737 006470 001110      MOV      @#TST4,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
3329 006526 012737 000004 001102      MOV      #4,@#$TSTNM      ;SET UP TEST NUMBER AND
                               ;CLEAR THE ERROR FLAG ($ERFLG)
3330                                ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3331 006534 013737 001102 177570      MOV      $TSTNM,@#DISPLAY
3332 006542 113737 002222 003524      MOVB     @#FS4,@#DPB.B+10 ;FS
3333 006550 113737 002220 003525      MOVB     @#FT4,@#DPB.B+11 ;FT
3334 006556 012737 007020 001270      MOV      @#EXIT4,@#BYPASS ;GO TO EXIT4 ON ERROR
3335 006564 005002          CLR      R2               ;CLEAR STALL SWITCH (NO STALL)
3336 006566 032737 010000 001246      BIT      @#SW12,@#C.SWR   ;STALL REQUIRED?
3337 006574 001401          BEQ     TEST4          ;NO--BRANCH
3338 006576 005102          COM     R2               ;YES--SET SWITCH
3339 006600 012706 001100          TEST4: MOV      @#STACK,SP      ;SET UP THE STACK POINTER
3340 006604 013701 002212          MOV      @#FC4,R1        ;SET NC TO FC
3341 006610 005037 001354          CLR     @#STALLO        ;START AT ZERO IF STALLS REQUIRED
3342 006614 010137 003526          1$:  MOV      R1,@#DPB.B+12   ;NC
3343 006620 004037 023032          JSR      RO,@#CALL.B      ;GO EXECUTE THE COMMAND

```

581
780

```

3344 006624 005702 TST R2 ;STALL?
3345 006626 001403 BEQ 2$ ;NO--BRANCH
3346 006630 004037 024144 JSR RO, @#STALL ;YES--GO TO STALL ROUTINE
3347 006634 001354 .WORD STALLO ;TIME POINTER
3348 006636 013737 002212 003526 2$: MOV FC4, @#DPB.B+12 ;FC
3349 006644 004037 023032 JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
3350 006650 005702 TST R2 ;STALL?
3351 006652 001413 BEQ 3$ ;NO--BRANCH
3352 006654 004037 024144 JSR RO, @#STALL ;YES--GO TO STALL ROUTINE
3353 006660 001354 .WORD STALLO ;TIME POINTER
3354 006662 005237 001354 INC @#STALLO ;UPDATE THE TIME
3355 006666 023737 001424 001354 CMP @#MXSTAL, @#STALLO ;TIME TO BIG?
3356 006674 003347 BGT 1$ ;NO--BRANCH
3357 006676 005037 001354 CLR @#STALLO ;YES--START OVER AT ZERO
3358 006702 063701 002216 3$: ADD @#IC4, R1 ;MOVE TO NEXT CYLINDER
3359 006706 020137 002214 CMP R1, @#LC4 ;LAST CYLINDER COMPLETED?
3360 006712 003740 BLE 1$ ;NO--BRANCH
3361 006714 013701 002214 MOV @#LC4, R1 ;SET NC TO LC
3362 006720 010137 003526 4$: MOV R1, @#DPB.B+12 ;NC
3363 006724 004037 023032 JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
3364 006730 005702 TST R2 ;STALL?
3365 006732 001403 BEQ 5$ ;NO--BRANCH
3366 006734 004037 024144 JSR RO, @#STALL ;YES--GO TO STALL ROUTINE
3367 006740 001354 .WORD STALLO ;TIME POINTER
3368 006742 013737 002214 003526 5$: MOV @#LC4, @#DPB.B+12 ;LC
3369 006750 004037 023032 JSR RO, @#CALL.B ;GO EXECUTE THE COMMAND
3370 006754 005702 TST R2 ;STALL?
3371 006756 001413 BEQ 6$ ;NO--BRANCH
3372 006760 004037 024144 JSR RO, @#STALL ;YES--GO TO STALL ROUTINE
3373 006764 001354 .WORD STALLO ;TIME POINTER
3374 006766 005237 001354 INC @#STALLO ;UPDATE STALL TIME
3375 006772 023737 001424 001354 CMP @#MXSTAL, @#STALLO ;TIME TOO BIG?
3376 007000 003347 BGT 4$ ;NO--BRANCH
3377 007002 005037 001354 CLR @#STALLO ;YES--SET STALL TIME BACK TO ZERO
3378 007006 163701 002216 6$: SUB @#IC4, R1 ;NEXT CYLINDER
3379 007012 020137 002212 CMP R1, @#FC4 ;DONE?
3380 007016 002373 BGE 6$ ;NO--BRANCH
3381 007020 000004 EXIT4: SCOPE ;LOOP

```

```

*****
; *TEST 5 CONVERGING/DIVERGING SEEK TEST

```

```

; * THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
; * SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
; * BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
; * GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
; * LESS THAN THE INITIAL VALUE OF "NC1". AT THE COMPLETION OF
; * EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
; * INSURE PROPER OPERATION. "NC1" AND "NC2" DEFAULT TO
; * "FC" AND "LC" RESPECTIVELY.

```

```

*****

```

```

3396 007022 TST5: BIT @#BITS+(5*2), TSTNMS ;DO THIS TEST?
3397 007022 033737 001442 001260 BNE 64$ ;YES--BRANCH
3398 007030 001002 JMP TST6 ;NO--GO TO THE NEXT TEST
3399 007032 000137 007200

```



```

3400 007036 013737 002226 001176 64$: MOV      @#RPT5,$TIMES ;GET THE ITERATION COUNT
3401 007044 012737 007116 001106      MOV      @#TEST5,@#SLPADR
3402 007052 012737 007022 001110      MOV      @#TST5,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
3403 007060 012737 000005 001102      MOV      #5,@#$TSTNM      ;SET UP TEST NUMBER AND
3404                                     ;CLEAR THE ERROR FLAG ($ERFLG)
3405 007066 013737 001102 177570      MOV      $TSTNM,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3406 007074 113737 002240 003524      MOV      @#FS5,@#DPB.B+10 ;FS
3407 007102 113737 002236 003525      MOV      @#FT5,@#DPB.B+11 ;FT
3408 007110 012737 007176 001270      MOV      @#EXIT5,@#BYPASS ;GO TO EXITS ON ERROR
3409 007116 012706 001100      TEST5: MOV      @#STACK,SP
3410 007122 013701 002230      MOV      @#FC5,R1 ;START NC1 AT FC
3411 007126 013702 002232      MOV      @#LC5,R2 ;START NC2 AT LC
3412 007132 010137 003526      1$: MOV      R1,@#DPB.B+12 ;NC1
3413 007136 004037 023032      JSR      RO,@#CALL.B ;GO EXECUTE THE COMMAND
3414 007142 010237 003526      MOV      R2,@#DPB.B+12 ;NC2
3415 007146 004037 023032      JSR      RO,@#CALL.B ;GO EXECUTE THE COMMAND
3416 007152 063701 002234      ADD      @#IC5,R1 ;NEXT NC1
3417 007156 163702 002234      SUB      @#IC5,R2 ;NEXT NC2
3418 007162 020137 002232      CMP      R1,@#LC5 ;DONE?
3419 007166 003003      BGT      EXITS ;YES--BRANCH
3420 007170 020237 002230      CMP      R2,@#FC5 ;?
3421 007174 002356      BGE      1$ ;NO--BRANCH
3422 007176 000004      EXITS: SCOPE ;LOOP

```

```

*****
*TEST 6 SERVO ADDRESSING LOGIC NOISE GENERATOR
*
* IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
* NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5. NOW "NC" IS UPDATED
* BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNTIL "LC" IS
* EXCEEDED BY ANY OF THE ABOVE VALUES. THE INITIAL VALUE OF "NC"
* IS "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE
* PROPER INDICATORS ARE EXAMINED TO INSURE PROPER OPERATION.

```

```

3435 007200      TST6: BIT      @#BITS+(6*2),TSTNMS ;DO THIS TEST?
3436 007200 033737 001444 001260      BNE      64$ ;YES--BRANCH
3437 007206 001002      JMP      TST7 ;NO--GO TO THE NEXT TEST
3438 007210 000137 007422      64$: MOV      @#RPT6,$TIMES ;GET THE ITERATION COUNT
3439 007214 013737 002244 001176      MOV      @#TEST6,@#SLPADR
3440 007222 012737 007274 001106      MOV      @#TST6,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
3441 007230 012737 007200 001110      MOV      #6,@#$TSTNM      ;SET UP TEST NUMBER AND
3442 007236 012737 000006 001102      MOV                                     ;CLEAR THE ERROR FLAG ($ERFLG)
3443                                     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3444 007244 013737 001102 177570      MOV      $TSTNM,@#DISPLAY
3445 007252 113737 002256 003524      MOV      @#FS6,@#DPB.B+10 ;FS
3446 007260 113737 002254 003525      MOV      @#FT6,@#DPB.B+11 ;FT
3447 007266 012737 007420 001270      MOV      @#EXIT6,@#BYPASS ;GO TO EXIT6 ON ERROR
3448 007274 012706 001100      TEST6: MOV      @#STACK,SP ;SETUP STACK
3449 007300 013701 002246      MOV      @#FC6,R1 ;PICKUP "FC"
3450 007304 013702 002250      MOV      @#LC6,R2 ;FORM LAST CYLINDER THAT
3451 007310 162702 000005      SUB      #5,R2 ;IS AVAILABLE FOR TESTING
3452 007314 020102      1$: CMP      R1,R2 ;LAST CYLINDER
3453 007316 003040      BGT      EXIT6 ;YES--BRANCH
3454 007320 010137 003526      MOV      R1,@#DPB.B+12 ;NC
3455 007324 004037 023032      JSR      RO,@#CALL.B ;GO EXECUTE THE COMMAND

```

3456	007330	062737	000004	003526	ADD	#4, @#DPB.B+12	;NC+4	
3457	007336	004037	023032		JSR	RD, @#CALL.B		;GO EXECUTE THE COMMAND
3458	007342	162737	000003	003526	SUB	#3, @#DPB.B+12	;NC+1	
3459	007350	004037	023032		JSR	RD, @#CALL.B		;GO EXECUTE THE COMMAND
3460	007354	062737	000002	003526	ADD	#2, @#DPB.B+12	;NC+3	
3461	007362	004037	023032		JSR	RD, @#CALL.B		;GO EXECUTE THE COMMAND
3462	007366	162737	000001	003526	SUB	#1, @#DPB.B+12	;NC+2	
3463	007374	004037	023032		JSR	RD, @#CALL.B		;GO EXECUTE THE COMMAND
3464	007400	062737	000003	003526	ADD	#3, @#DPB.B+12	;NC+5	
3465	007406	004037	023032		JSR	RD, @#CALL.B		;GO EXECUTE THE COMMAND
3466	007412	063701	002252		ADD	@#IC6, R1		
3467	007416	000736			BR	IS		
3468	007420	000004			EXIT6:	SCOPE		;LOOP

3469
3470
3471

.SBTTL *** TIMING TESTS (7-12) ***

;/\:
;/\:
;/\:

;*THE TIMING TESTS WILL INSURE THAT THOSE FUNCTIONS BEING
;*TIMED ARE WITHIN THE TOLERANCES SPECIFIED IN THE "RPO4
;*ENGINEERING SPECIFICATIONS".
;*THE SEEK TIMING WILL BE PERFORMED USING EXPLICIT SEEK
;*OPERATIONS. AT THE COMPLETION OF EACH OF THE TIMING
;*TESTS THE MINIMUM, MAXIMUM AND AVERAGE TIMES WILL BE
;*TYPED.

;/\:
;/\:
;/\:

3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524

*TEST 7 ROTATIONAL SPEED TIMING TEST

;* THIS TEST WILL START A SEARCH TO CYLINDER 0, TRACK 0, SECTOR
;* 0. AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS SET AGAIN
;* AND THE OPERATION IS TIMED. THIS PROCEDURE IS REPEATED 10
;* TIMES THEN THE AVERAGE TIME IS CALCULATED AND CHECKED TO
;* INSURE IT IS WITHIN TOLERANCE:
;* 16.67 MS/REV + OR - 2% IF 60HZ
;* 16.67 MS/REV + OR - 2.5% IF 50HZ.

TST7:

007422	033737	001446	001260	BIT	Q#BITS+(<7*2),TSTNMS ;DO THIS TEST?
007422	001002			BNE	64\$;YES--BRANCH
007432	000137	010142		JMP	TST10 ;NO--GO TO THE NEXT TEST
007436	013737	002262	001176	64\$: MOV	Q#RPT7,\$TIMES ;GET THE ITERATION COUNT
007444	012737	007422	001106	MOV	#TST7,Q#SLPADR ;SETUP THE SCOPE LOOP ADDRESS
007452	012737	007422	001110	MOV	#TST7,Q#SLPERR ;SETUP THE ERROR LOOP ADDRESS
007460	012737	000007	001102	MOV	#7,Q#\$TSTNM ;SET UP TEST NUMBER AND
007466	013737	001102	177570	MOV	\$TSTNM,Q#DISPLAY ;CLEAR THE ERROR FLAG (SERFLG)
007474	005737	001262		TST	Q#CLKSTA ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
007500	003002			BGT	1\$;KW11-P CLOCK?
007502	000137	010142		JMP	TST10 ;YES--START TEST
007506	004037	024416		1\$: JSR	RO,Q#SRCH00 ;NO--GO TO NEXT TEST
007512	000401			BR	2\$;DO A MASSBUS INIT & RECAL
007514	000563			BR	EXIT7 ;RETURN HERE IF NO ERROR
007516	013764	002264	000034	2\$: MOV	Q#FC7,RHCA(R4) ;RETURN HERE IF ERROR
007524	013746	002270		MOV	Q#FS7,-(SP) ;FC
007530	113766	002266	000001	MOV	Q#FT7,1(SP) ;FS
007536	012664	000006		MOV	(SP)+,RHDA(R4) ;FT
007542	012737	010064	001200	MOV	#EXIT7,\$ESCAPE ;LOAD FT/FS
007550	005005			CLR	R5 ;ESCAPE TO EXIT7 ON ERROR
007552	012703	002364		MOV	#T7A,R3 ;COUNT UP
					60HZ PARAMETERS

3525	007556	032737	000100	001246	BIT	#SW06, @#C.SWR	:60 HZ?
3526	007564	001402			BEQ	TEST7	:YES--BRANCH
3527	007566	012703	002374		MOV	#T7B, R3	:NO--50 HZ PARAMETERS
3528	007572	012706	001100		TEST7: MOV	#STACK, SP	:SETUP STACK
3529	007576	012701	000012		MOV	#D10, R1	:TIME 10 SEARCHES
3530	007602	004737	024602		JSR	PC, @#STRMTR	:INITIALIZE THE TIMERS
3531	007606	012777	010014	171554	MOV	#7\$, @PKV	:SETUP VECTOR IN CASE OF OVERFLOW
3532	007614	012777	024600	022540	MOV	#DORTI, @RPVEC	:SETUP RPO4 VECTOR
3533	007622	005077	171550		1\$: CLR	@PKB	:START COUNTING AT ZERO
3534	007626	012777	000131	171540	MOV	#131, @PKCS	:INT.EN., COUNT UP AT 100KHZ
3535	007634	012714	000131		MOV	#SEARCH, (R4)	:START A SEARCH
3536	007640	000001			WAIT		:WAIT ON INTERRUPT
3537	007642	042777	000101	171524	BIC	#101, @PKCS	:STOP THE CLOCK
3538	007650	032764	040000	000012	BIT	#BIT14, RHDS1(R4)	:ERROR?
3539	007656	001415			BEQ	2\$:NO--BRANCH
3540	007660	104416			SAVREG		:SAVE R0-R5
3541	007662	012702	003554		MOV	#DTADPB, R2	:DPB POINTER
3542	007666	004737	037044		JSR	PC, @#SVRH11	:SAVE ALL THE RH11/RPO4 REGISTERS
3543	007672	012764	000040	000010	MOV	#BIT05, RHCS2(R4)	:MASSBUS CLEAR
3544	007700	013764	003554	000010	MOV	@#DTADPB, RHCS2(R4)	:SELECT DRIVE
3545	007706	104420			RESREG		:RESTORE R0-R5
3546	007710	104017			ERROR	17	
3547	007712	005077	171460		2\$: CLR	@PKB	:START THE COUNT AT ZERO
3548	007716	012714	000131		MOV	#SEARCH, (R4)	:START A SEARCH
3549	007722	012777	000131	171444	MOV	#131, @PKCS	:START THE CLOCK
3550	007730	000001			WAIT		:WAIT ON INTERRUPT
3551	007732	042777	000101	171434	BIC	#101, @PKCS	:STOP THE CLOCK
3552	007740	032764	040000	000012	BIT	#BIT14, RHDS1(R4)	:IS "ERR=1"?
3553	007746	001415			BEQ	3\$:NO--BRANCH
3554	007750	104416			SAVREG		:SAVE R0-R5
3555	007752	012702	003554		MOV	#DTADPB, R2	:DPB POINTER
3556	007756	004737	037044		JSR	PC, @#SVRH11	:SAVE ALL THE RH11/RPO4 REGISTERS
3557	007762	012764	000040	000010	MOV	#BIT05, RHCS2(R4)	:MASSBUS CLEAR
3558	007770	013764	003554	000010	MOV	@#DTADPB, RHCS2(R4)	:SELECT DRIVE
3559	007776	104420			RESREG		:RESTORE R0-R5
3560	010000	104017			ERROR	17	:DISK ERROR OCCURRED
3561	010002	004737	024646		3\$: JSR	PC, @#COUNT	:UPDATE THE COUNT
3562	010006	005301			DEC	R1	:DONE?
3563	010010	003304			BGT	1\$:NO--BRANCH
3564	010012	000424			BR	EXIT7	:YES--GO TO THE EXIT
3565	010014	042777	000101	171352	7\$: BIC	#101, @PKCS	:STOP THE CLOCK
3566	010022	005037	177776		CLR	@#PS	:DROP THE PRIORITY
3567	010026	012600			MOV	(SP)+, R0	:PC OF WAIT+2
3568	010030	005726			TST	(SP)+	:POP THE PS FROM THE STACK
3569	010032	104416			SAVREG		:SAVE R0-R5
3570	010034	012702	003554		MOV	#DTADPB, R2	:DPB POINTER
3571	010040	004737	037044		JSR	PC, @#SVRH11	:SAVE ALL THE RH11/RPO4 REGISTERS
3572	010044	012764	000040	000010	MOV	#BIT05, RHCS2(R4)	:MASSBUS CLEAR
3573	010052	013764	003554	000010	MOV	@#DTADPB, RHCS2(R4)	:SELECT DRIVE
3574	010060	104420			RESREG		:RESTORE R0-R5
3575	010062	104020			ERROR	20	:CLOCK OVERFLOWED
3576	010064	000004			EXIT7: SCOPE		
3577	010066	012764	000040	000010	MOV	#BIT05, RHCS2(R4)	:MASSBUS INIT.
3578	010074	013764	003554	000010	MOV	@#DTADPB, RHCS2(R4)	:SELECT DRIVE
3579	010102	004737	022402		JSR	PC, @#ST.CLK	:INITIALIZE THE CLOCK
3580	010106	012777	034766	022246	MOV	#ISR, @RPVEC	:RESTORE RH11/RPO4 INT. VECTOR


```

3581 010114 032737 000100 001246 BIT #SW06,@#C.SWR ;60 HZ?
3582 010122 001004 BNE 1$ ;NO -- BRANCH
3583 010124 004037 025000 JSR RO,@#TYPTIM ;GO TYPE THE TIMES
3584 010130 002364 T7A ;POINTER
3585 010132 000403 BR TST10 ;GO TO NEXT TEST
3586 010134 1$: JSR RO,@#TYPTIM ;GO TYPE THE TIMES
3587 010134 004037 025000 T7B ;POINTER
3588 010140 002374

```

```

3589
3590
3591 ;*****
3592 ;*TEST 10 ONE CYLINDER SEEK TIMING TEST
3593

```

```

3594 ;* THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
3595 ;* CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
3596 ;* NUMBER OF CYLINDERS TO CYLINDER 410, THEN REVERSE SEEK
3597 ;* CYCLES TO RETRACT THE CYLINDER BY ONE UNTIL THE INCREMENT
3598 ;* IS GREATER THAN THE NUMBER OF CYLINDERS TO CYLINDER 0. THE
3599 ;* TIME TO PERFORM EACH SEEK IS CHECKED TO INSURE IT DOES NOT
3600 ;* EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
3601 ;* THE TIME MUST BE LESS THAN 10MS BUT GREATER THAN 3MS.
3602

```

```

3603 ;*****
3604 TST10:
3605 010142 033737 001450 001260 BIT @#BITS+<10*2>,TSTNMS ;DO THIS TEST?
3606 010150 001002 BNE 64$ ;YES--BRANCH
3607 010152 000137 010564 JMP TST11 ;NO--GO TO THE NEXT TEST
3608 010156 013737 002274 001176 64$: MOV @#RPT10,$TIMES ;GET THE ITERATION COUNT
3609 010164 012737 010142 001106 MOV #TST10,@#$LPADR ;SETUP THE SCOPE LOOP ADDRESS
3610 010172 012737 010142 001110 MOV #TST10,@#$LPERR ;SETUP THE ERROR LOOP ADDRESS
3611 010200 012737 000010 001102 MOV #10,@#$TSTNM ;SET UP TEST NUMBER AND
3612 ; CLEAR THE ERROR FLAG ($ERFLG)
3613 010206 013737 001102 177570 MOV $TSTNM,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3614 010214 005737 001262 TST @#CLKSTA ;KW11-P CLOCK?
3615 010220 003002 BGT 1$ ;YES--START TEST
3616 010222 000137 010564 JMP TST11 ;NO--GO TO NEXT TEST
3617 010226 004037 024416 1$: JSR RO,@#SRCH00 ;DO A MASSBUS INIT. AND RECAL
3618 010232 000401 BR 2$ ;NO ERROR RETURN
3619 010234 000534 BR EXIT10 ;ERROR RETURN--SCOPE LOOP CALL
3620 010236 012703 002404 2$: MOV #T10,R3 ;PARAMETER POINTER
3621 010242 012737 010526 001200 MOV #EXIT10,$ESCAPE ;ESCAPE TO EXIT10 ON ERROR
3622 010250 012706 001100 TEST10: MOV #STACK,$P ;SETUP STACK
3623 010254 012737 000001 003566 MOV #1,@#DTADPB+12 ;START WITH CYLINDER #1
3624 010262 005005 CLR R5 ;SET THE UP/DOWN SWITCH TO UP
3625 010264 004737 024602 JSR PC,@#STRTMR ;INITIALIZE THE TIMERS
3626 010270 012777 010456 171072 MOV #7,$@PKV ;SETUP INCASE OF OVERFLOW
3627 010276 012777 024600 022056 MOV #DORTI,@RPVEC ;SET RPO4 VECTOR
3628 010304 005077 171066 1$: CLR @PKB ;START THE COUNTER AT ZERO
3629 010310 013764 003566 000034 MOV @#DTADPB+12,RHCA(R4) ;LOAD DESIRED CYLINDER
3630 010316 012714 000105 MOV #SEEK,(R4) ;START A SEEK
3631 010322 012777 000131 171044 MOV #131,@PKCS ;START THE CLOCK
3632 010330 000001 WAIT ;WAIT ON INTERRUPT
3633 010332 042777 000101 171034 BIC #101,@PKCS ;STOP THE CLOCK
3634 010340 032764 040000 000012 BIT #BIT14,RHDS1(R4) ;ANY DISK ERRORS?
3635 010346 001415 BEQ 2$ ;NO--BRANCH
3636 010350 104416 SAVREG ;SAVE R0-R5

```



```

3637 010352 012702 003554      MOV      #DTADPB,R2      ;DPB POINTER
3638 010356 004737 037044      JSR      PC,@#SVRH11    ;SAVE ALL THE RH11/RP04 REGISTERS
3639 010362 012764 000040 000010  MOV      #BIT05,RHCS2(R4) ;MASSBUS CLEAR
3640 010370 013764 003554 000010  MOV      @#DTADPB,RHCS2(R4) ;SELECT DRIVE
3641 010376 104420      RESREG      ;RESTORE R0-R5
3642 010400 104017      ERROR      17          ;REPORT THE ERROR
3643 010402 004737 024646      JSR      PC,@#COUNT    ;COUNT THIS SEEKS TIME
3644 010406 004737 024226      JSR      PC,@#TWOMS     ;STALL FOR 2 MILLISECONDS
3645 010412 005705      TST      R5            ;UP OR DOWN?
3646 010414 001011      BNE      4$           ;DOWN--BRANCH
3647 010416 005237 003566      INC      @#DTADPB+12    ;MOVE TO NEXT CYLINDER
3648 010422 023727 003566 000632  CMP      @#DTADPB+12, #10 ;OUT OF CYLINDERS?
3649 010430 002725      BLT      1$           ;NO--GO DO THE NEXT SEEK
3650 010432 012705 177777      MOV      #-1,R5        ;SET UP/DOWN SWITCH TO DOWN
3651 010436 000722      BR       1$           ;GO DO THE NEXT SEEK
3652 010440 005337 003566      DEC      @#DTADPB+12    ;MOVE TO NEXT CYLINDER
3653 010444 023727 003566 000000  CMP      @#DTADPB+12, #0 ;OUT OF CYLINDERS?
3654 010452 003314      BGT      1$           ;NO--GO DO THE NEXT SEEK
3655 010454 000424      BR       EXIT10       ;GO TO THE EXIT
3656 010456 042777 000101 170710  BIC      #101,@PKCS     ;STOP THE CLOCK
3657 010464 005037 177776      CLR      @#PS          ;DROP THE PRIORITY
3658 010470 012600      MOV      (SP)+,R0      ;PC OF WAIT+2
3659 010472 005726      TST      (SP)+        ;POP THE PS FROM THE STACK
3660 010474 104416      SAVREG      ;SAVE R0-R5
3661 010476 012702 003554      MOV      #DTADPB,R2      ;DPB POINTER
3662 010502 004737 037044      JSR      PC,@#SVRH11    ;SAVE ALL THE RH11/RP04 REGISTERS
3663 010506 012764 000040 000010  MOV      #BIT05,RHCS2(R4) ;MASSBUS CLEAR
3664 010514 013764 003554 000010  MOV      @#DTADPB,RHCS2(R4) ;SELECT DRIVE
3665 010522 104420      RESREG      ;RESTORE R0-R5
3666 010524 104020      ERROR      20          ;REPORT CLOCK OVERFLOW
3667 010526 000004      EXIT10: SCOPE
3668 010530 012764 000040 000010  MOV      #BIT05,RHCS2(R4) ;MASSBUS INIT.
3669 010536 013764 003554 000010  MOV      @#DTADPB,RHCS2(R4) ;SELECT DRIVE
3670 010544 004737 022402      JSR      PC,@#ST.CLK    ;INITIALIZE THE CLOCK
3671 010550 012777 034766 021604  MOV      #ISR,@RPVEC    ;RESTORE RH11/RP04 INT. VECTOR
3672 010556 004037 025000      JSR      R0,@#TYPTIM   ;GO TYPE THE TIMES
3673 010562 002404      T10          ;POINTER

```

```

3674
3675
3676 ;*****
3677 ;*TEST 11 AVERAGE SEEK TIMING TEST
3678
3679 ;* THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
3680 ;* CYLINDER 136, THEN A REVERSE SEEK FROM CYLINDER 136 TO
3681 ;* CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO INSURE THEY
3682 ;* ARE WITHIN THE TOLERANCE ALLOWED FOR THE AVERAGE SEEK TIME.
3683 ;* THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL
3684 ;* OF 256 SEEKS). THE AVERAGE SEEK TIME IS 28 MS + OR - 2MS.
3685
3686 ;*****
3687 †TST11:
3688 010564 033737 001452 001260 BIT @#BITS+<11*2>,TSTNMS ;DO THIS TEST?
3689 010572 001002 BNE 64$ ;YES--BRANCH
3690 010574 000137 011264 JMP TST12 ;NO--GO TO THE NEXT TEST
3691 010600 013737 002300 001176 64$: MOV @#RPT11,$TIMES ;GET THE ITERATION COUNT
3692 010606 012737 010564 001106 MOV #TST11,@#SLPADR ;SETUP THE SCOPE LOOP ADDRESS

```

Handwritten initials or mark.

3693	010614	012737	010564	001110		MOV	#TST11, 2#SLPERR ;	SETUP THE ERROR LOOP ADDRESS
3694	010622	012737	000011	001102		MOV	#11, 2#STSTNM ;	SET UP TEST NUMBER AND
3695								CLEAR THE ERROR FLAG (SERFLG)
3696	010630	013737	001102	177570		MOV	STSTNM, 2#DISPLAY ;	LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3697	010636	005737	001262			TST	2#CLKSTA ;	KW11-P CLOCK?
3698	010642	003002				BGT	1\$;	YES--START TEST
3699	010644	000137	011264			JMP	TST12 ;	NO--GO TO NEXT TEST
3700	010650	004037	024416		1\$:	JSR	RO, 2#SRCH00 ;	DO A MASSBUS INIT & RECAL
3701	010654	000401				BR	2\$;	RETURN HERE IF NO ERROR
3702	010656	000563				BR	EXIT11 ;	RETURN HERE ON ERROR
3703	010660	012703	002414		2\$:	MOV	#T11, R3 ;	PARAMETER POINTER
3704	010664	012737	011226	001200		MOV	#EXIT11, \$ESCAPE ;	ESCAPE TO EXIT11 ON ERROR
3705	010672	012706	001100		TEST11:	MOV	#STACK, SP ;	SETUP STACK
3706	010676	012701	000200			MOV	#D128, R1 ;	REPEAT "0-136-0" 128 TIMES
3707	010702	004737	024602			JSR	PC, 2#STRTMR ;	INIT. THE COUNTERS
3708	010706	012777	011156	170454		MOV	#7\$, 2#PKV ;	SET UP VECTOR IN CASE OF OVERFLOW
3709	010714	012777	024600	021440		MOV	#DORT1, 2#RVEC ;	SETUP RPO4 VECTOR
3710	010722	005077	170450		1\$:	CLR	2#PKB ;	START COUNT AT ZERO
3711	010726	012764	000210	000034		MOV	#D136, RHCA(R4) ;	CYLINDER=136
3712	010734	012764	000105	000000		MOV	#SEEK, RHCS1(R4) ;	START A SEEK
3713	010742	012777	000131	170424		MOV	#131, 2#PKCS ;	START THE CLOCK
3714	010750	000001				WAIT		WAIT ON INTERRUPT
3715	010752	042777	000101	170414		BIC	#101, 2#PKCS ;	STOP CLOCK
3716	010760	032764	040000	000012		BIT	#BIT14, RHDS1(R4) ;	ERR=1?
3717	010766	001415				BEQ	2\$;	NO--BRANCH
3718	010770	104416				SAVREG		SAVE R0-R5
3719	010772	012702	003554			MOV	#DTADPB, R2 ;	DPB POINTER
3720	010776	004737	037044			JSR	PC, 2#SVRH11 ;	SAVE ALL THE RH11/RPO4 REGISTERS
3721	011002	012764	000040	000010		MOV	#BIT05, RHCS2(R4) ;	MASSBUS CLEAR
3722	011010	013764	003554	000010		MOV	2#DTADPB, RHCS2(R4) ;	SELECT DRIVE
3723	011016	104420				RESREG		RESTORE R0-R5
3724	011020	104017				ERROR	17	
3725	011022	005005			2\$:	CLR	R5 ;	SET UP/DOWN SWITCH TO UP
3726	011024	004737	024646			JSR	PC, 2#COUNT ;	UPDATE THE COUNT
3727	011030	004737	024226			JSR	PC, 2#TWOMS ;	STALL FOR 2 MILLISECONDS
3728	011034	005077	170336			CLR	2#PKB ;	START THE COUNT AT ZERO
3729	011040	012764	000000	000034		MOV	#0, RHCA(R4) ;	CYLINDER=0
3730	011046	012764	000105	000000		MOV	#SEEK, RHCS1(R4) ;	START A SEEK
3731	011054	012777	000131	170312		MOV	#131, 2#PKCS ;	START THE CLOCK
3732	011062	000001				WAIT		WAIT ON INTERRUPT
3733	011064	042777	000101	170302		BIC	#101, 2#PKCS ;	STOP THE CLOCK
3734	011072	032764	040000	000012		BIT	#BIT14, RHDS1(R4) ;	ERR=1?
3735	011100	001415				BEQ	3\$;	NO--BRANCH
3736	011102	104416				SAVREG		SAVE R0-R5
3737	011104	012702	003554			MOV	#DTADPB, R2 ;	DPB POINTER
3738	011110	004737	037044			JSR	PC, 2#SVRH11 ;	SAVE ALL THE RH11/RPO4 REGISTERS
3739	011114	012764	000040	000010		MOV	#BIT05, RHCS2(R4) ;	MASSBUS CLEAR
3740	011122	013764	003554	000010		MOV	2#DTADPB, RHCS2(R4) ;	SELECT DRIVE
3741	011130	104420				RESREG		RESTORE R0-R5
3742	011132	104017				ERROR	17	
3743	011134	012705	177777		3\$:	MOV	#-1, R5 ;	SET UP/DOWN SWITCH TO DOWN
3744	011140	004737	024646			JSR	PC, 2#COUNT ;	UPDATE THE COUNT
3745	011144	004737	024226			JSR	PC, 2#TWOMS ;	STALL FOR 2 MILLISECONDS
3746	011150	005301				DEC	R1 ;	DONE?
3747	011152	003263				BGT	1\$;	NO--BRANCH
3748	011154	000424				BR	EXIT11 ;	YES--EXIT


```

3749 011156 042777 000101 170210 7$: BIC #101,DPKCS ;STOP THE CLOCK
3750 011164 005037 177776 CLR @PS ;DROP THE PRIORITY
3751 011170 012600 MOV (SP)+,RO ;PC OF WAIT+2
3752 011172 005726 TST (SP)+ ;POP THE PS FROM THE STACK
3753 011174 104416 SAVREG ;SAVE RO-R5
3754 011176 012702 003554 MOV #DTADPB,R2 ;DPB POINTER
3755 011202 004737 037044 JSR PC,@SVRH11 ;SAVE ALL THE RH11/RP04 REGISTERS
3756 011206 012764 000040 000010 MOV #BIT05,RHCS2(R4) ;MASSBUS CLEAR
3757 011214 013764 003554 000010 MOV @DTADPB,RHCS2(R4) ;SELECT DRIVE
3758 011222 104420 RESREG ;RESTORE RO-R5
3759 011224 104020 ERROR 20 ;CLOCK OVERFLOWED
3760 011226 000004 EXIT11: SCOPE
3761 011230 012764 000040 000010 MOV #BIT05,RHCS2(R4) ;MASSBUS INIT.
3762 011236 013764 003554 000010 MOV @DTADPB,RHCS2(R4) ;SELECT DRIVE
3763 011244 004737 022402 JSR PC,@ST.CLK ;INITIALIZE THE CLOCK
3764 011250 012777 034766 021104 MOV #ISR,@RPVEC ;RESTORE RH11/RP04 INT. VECTOR
3765 011256 004037 025000 JSR RO,@TYPTIM ;GO TYPE THE TIMES
3766 011262 002414 T11 ;POINTER

```

```

*****
*TEST 12 MAXIMUM SEEK TIMING TEST

```

```

* THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
* CYLINDER 410, THEN A REVERSE SEEK FROM CYLINDER 410 TO
* CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO INSURE
* THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
* TIME. THIS SEQUENCE IS REPEATED 128 TIMES (FOR
* A TOTAL OF 256 SEEKS). THE MAXIMUM SEEK TIME IS 50MS + 4MS OR - 2MS.

```

```

*****
TST12:

```

```

3779 011264 033737 001454 001260 BIT @#BITS+(12*2),TSTNMS ;DO THIS TEST?
3780 011264 001002 BNE 64$ ;YES--BRANCH
3781 011272 000137 011764 JMP TST13 ;NO--GO TO THE NEXT TEST
3782 011274 000137 011764 64$: MOV @RPT12,$TIMES ;GET THE ITERATION COUNT
3783 011300 013737 002304 001176 64$: MOV @TST12,@$SLPADR ;SETUP THE SCOPE LOOP ADDRESS
3784 011306 012737 011264 001106 64$: MOV @TST12,@$SLPERR ;SETUP THE ERROR LOOP ADDRESS
3785 011314 012737 011264 001110 64$: MOV #12,@$TSTNM ;SET UP TEST NUMBER AND
3786 011322 012737 000012 001102 64$: ;CLEAR THE ERROR FLAG (SERFLG)
3787 ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3788 011330 013737 001102 177570 MOV $TSTNM,@DISPLAY
3789 011336 005737 001262 TST @CLKSTA ;KW11-P CLOCK
3790 011342 003002 BGT 1$ ;YES--START TEST
3791 011344 000137 011764 JMP TST13 ;NO--GO TO NEXT TEST
3792 011350 004037 024416 1$: JSR RO,@SRCH00 ;DO A MASSBUS INIT & RECAL
3793 011354 000401 BR 2$ ;RETURN HERE IF NO ERROR
3794 011356 000563 BR EXIT12 ;RETURN HERE ON ERROR
3795 011360 012703 002424 2$: MOV #T12,R3 ;PARAMETER POINTER
3796 011364 012737 011726 001200 2$: MOV #EXIT12,$ESCAPE ;ESCAPE TO EXIT12 ON ERROR
3797 011372 012706 001100 TEST12: MOV #STACK,$P ;SETUP STACK
3798 011376 012701 000200 MOV #D128,R1 ;REPEAT "0-410-0" 128 TIMES
3799 011402 004737 024602 JSR PC,@$STMR ;INIT. THE TIMERS
3800 011406 012777 011656 167754 MOV #7,$@PKV ;SETUP VECTOR IN CASE OF OVERFLOW
3801 011414 012777 024600 020740 MOV #DORTI,@RPVEC ;SETUP RP04 VECTOR
3802 011422 005077 167750 1$: CLR @PKB ;START COUNTING FROM ZERO
3803 011426 012764 000632 000034 1$: MOV #410.,RHCA(R4) ;CYLINDER=410
3804 011434 012764 000105 000000 1$: MOV #SEEK,RHCS1(R4) ;START A SEEK

```


3805	011442	012777	000131	167724	MOV	#131,DPKCS	;START THE CLOCK
3806	011450	000001			WAIT		;WAIT ON INTERRUPT
3807	011452	042777	000101	167714	BIC	#101,DPKCS	;STOP THE CLOCK
3808	011460	032764	040000	000012	BIT	#BIT14,RHDS1(R4)	;ERR=1?
3809	011466	001415			BEQ	25	;NO--BRANCH
3810	011470	104416			SAVREG		;SAVE R0-R5
3811	011472	012702	003554		MOV	#DTADPB,R2	;DPB POINTER
3812	011476	004737	037044		JSR	PC,#SVRH11	;SAVE ALL THE RH11/RP04 REGISTERS
3813	011502	012764	000040	000010	MOV	#BIT05,RHCS2(R4)	;MASSBUS CLEAR
3814	011510	013764	003554	000010	MOV	#DTADPB,RHCS2(R4)	;SELECT DRIVE
3815	011516	104420			RESREG		;RESTORE R0-R5
3816	011520	104017			ERROR	17	
3817	011522	005005			CLR	R5	;SET THE UP/DOWN SWITCH TO UP
3818	011524	004737	024646		JSR	PC,#COUNT	;UP THE COUNT
3819	011530	004737	024226		JSR	PC,#TWOMS	;STALL FOR 2 MILLISECONDS
3820	011534	005077	167636		CLR	DPKB	;START COUNT AT ZERO
3821	011540	012764	000000	000034	MOV	#0,RHCA(R4)	;CYLINDER=0
3822	011546	012764	000105	000000	MOV	#SEEK,RHCS1(R4)	;START A SEEK
3823	011554	012777	000131	167612	MOV	#131,DPKCS	;START THE CLOCK
3824	011562	000001			WAIT		;WAIT ON INTERRUPT
3825	011564	042777	000101	167602	BIC	#101,DPKCS	;STOP THE CLOCK
3826	011572	032764	040000	000012	BIT	#BIT14,RHDS1(R4)	;ERR=1?
3827	011600	001415			BEQ	35	;NO--BRANCH
3828	011602	104416			SAVREG		;SAVE R0-R5
3829	011604	012702	003554		MOV	#DTADPB,R2	;DPB POINTER
3830	011610	004737	037044		JSR	PC,#SVRH11	;SAVE ALL THE RH11/RP04 REGISTERS
3831	011614	012764	000040	000010	MOV	#BIT05,RHCS2(R4)	;MASSBUS CLEAR
3832	011622	013764	003554	000010	MOV	#DTADPB,RHCS2(R4)	;SELECT DRIVE
3833	011630	104420			RESREG		;RESTORE R0-R5
3834	011632	104017			ERROR	17	;REPORT THE ERROR
3835	011634	012705	177777		MOV	#-1,R5	;SET THE UP/DOWN SWITCH TO DOWN
3836	011640	004737	024646		JSR	PC,#COUNT	;UPDATE THE COUNT
3837	011644	004737	024226		JSR	PC,#TWOMS	;STALL FOR 2 MILLISECONDS
3838	011650	005301			DEC	R1	;DONE?
3839	011652	003263			BGT	15	;NO--BRANCH
3840	011654	000424			BR	EXIT12	;YES--EXIT
3841	011656	042777	000101	167510	BIC	#101,DPKCS	;STOP THE CLOCK
3842	011664	005037	177776		CLR	#PS	;DROP THE PRIORITY
3843	011670	012600			MOV	(SP)+,R0	;PC OF WAIT+2
3844	011672	005726			TST	(SP)+	;POP THE PS FROM THE STACK
3845	011674	104416			SAVREG		;SAVE R0-R5
3846	011676	012702	003554		MOV	#DTADPB,R2	;DPB POINTER
3847	011702	004737	037044		JSR	PC,#SVRH11	;SAVE ALL THE RH11/RP04 REGISTERS
3848	011706	012764	000040	000010	MOV	#BIT05,RHCS2(R4)	;MASSBUS CLEAR
3849	011714	013764	003554	000010	MOV	#DTADPB,RHCS2(R4)	;SELECT DRIVE
3850	011722	104420			RESREG		;RESTORE R0-R5
3851	011724	104020			ERROR	20	;CLOCK OVERFLOWED
3852	011726	000004			EXIT12:	SCOPE	
3853	011730	012764	000040	000010	MOV	#BIT05,RHCS2(R4)	;MASSBUS INIT.
3854	011736	013764	003554	000010	MOV	#DTADPB,RHCS2(R4)	;SELECT DRIVE
3855	011744	004737	022402		JSR	PC,#ST.CLK	;INITIALIZE THE CLOCK
3856	011750	012777	034766	020404	MOV	#ISR,DPVEC	;RESTORE RH11/RP04 INT. VECTOR
3857	011756	004037	025000		JSR	R0,#TYPTIM	;GO TYPE THE TIMES
3858	011762	002424			T12		;POINTER

3859
3860
3861

.SBTTL *** ADDRESSING TESTS (13-14) ***

:/\:
:/\:
:/\:

;*THE ADDRESSING TESTS INSURES PROPER OPERATION OF THE TRACK
;*AND SECTOR ADDRESS CIRCUITRY. BOTH ADDRESSING TESTS
;*WILL BE PERFORMED ON CYLINDER FC.

:/\:
:/\:
:/\:

3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914

```
011764
011764 033737 001456 001260
011772 001002
011774 000137 012264
012000 013737 002310 001176
012006 012737 012044 001106
012014 012737 011764 001110
012022 012737 000013 001102
012030 013737 001102 177570
012036 012737 012262 001270
012044 012706 001100
012050 004737 025406
012054 012737 000161 003556
012062 013737 002312 003566
012070 113737 002314 003565
012076 105037 003564
012102 012737 165000 003560
012110 012737 044134 003562
012116 004037 023342
012122 012737 000151 003556
012130 004037 023342
012134 004037 025450
012140 012737 000171 003556
012146 004037 023342
012152 004037 025522
```

;*TEST 13 SECTOR ADDRESSING TEST

;* THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT". THE DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR BEING WRITTEN. A WRITE CHECK IS PERFORMED, THE BUFFER IS CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED. THEN SECTOR 0 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED. THEN SECTOR 1 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED. THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH REWRITE SECTOR 21 AND WRITE CHECK SECTORS 0-21.

*TST13:

```
BIT          @#BITS+<13*2>,TSTNMS ;DO THIS TEST?
BNE          64$                    ;YES--BRANCH
JMP          TST14                  ;NO--GO TO THE NEXT TEST
64$:         MOV          @#RPT13,S/TMES ;GET THE ITERATION COUNT
MOV          @#TEST13,@#$LADR
MOV          @#TST13,@#$LPERR ;SETUP THE ERROR LOOP ADDRESS
MOV          @#13,@#$TSTNM ;SET UP TEST NUMBER AND
                                ;CLEAR THE ERROR FLAG (SERFLG)
MOV          @#STNM,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV          @#EXIT13,@#BYPASS
TEST13:     MOV          @#STACK,SP ;SET THE STACK POINTER
JSR          PC,@#FILBUF ;FILL THE BUFFER WITH SECTOR ADDRESSES
MOV          @#WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
MOV          @#FC13,@#DTADPB+12 ;CYLINDER
MOV         @#FT13,@#DTADPB+11 ;TRACK
CLRB        @#DTADPB+10 ;SECTOR
MOV         @#TRCKWC,@#DTADPB+4 ;WORD COUNT
MOV         @#BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
JSR         RO,@#DRVCAL ;START A DATA TRANSFER
MOV         @#WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
JSR         RO,@#DRVCAL ;START A DATA TRANSFER
JSR         RO,@#CLRBUF ;CLEAR BUFFER
MOV         @#READ,@#DTADPB+2 ;COMMAND = READ
JSR         RO,@#DRVCAL ;START A DATA TRANSFER
JSR         RO,@#CKSCTR ;CHECK THE SECTOR DATA READ
```



```

3915 012156 012700 044134      MOV      #BUFFER,RO      ;BUFFER ADDRESS
3916 012162 005001              CLR      R1              ;FIRST SECTOR
3917 012164 012737 000161 003556 1$:  MOV      #WRITE,2#DTADPB+2 ;COMMAND=WRITE DATA
3918 012172 012737 177400 003560      MOV      #SCTRWC,2#DTADPB+4 ;WORD COUNT
3919 012200 010037 003562      MOV      RO,2#DTADPB+6    ;BUFFER ADDRESS
3920 012204 110137 003564      MOVVB   R1,2#DTADPB+10   ;SECTOR
3921 012210 004037 023342      JSR      RO,2#DRVCAL     ;START A DATA TRANSFER
3922 012214 012737 000151 003556      MOV      #WRCKD,2#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3923 012222 012737 165000 003560      MOV      #TRCKWC,2#DTADPB+4 ;WORD COUNT
3924 012230 012737 044134 003562      MOV      #BUFFER,2#DTADPB+6 ;BUFFER ADDRESS
3925 012236 105037 003564      CLRB    2#DTADPB+10     ;SECTOR
3926 012242 004037 023342      JSR      RO,2#DRVCAL     ;START A DATA TRANSFER
3927 012246 062700 001000      ADD      #10256*2,RO     ;MOVE TO NEXT SECTOR
3928 012252 005201              INC      R1
3929 012254 022701 000026      CMP      #1022,R1       ;DONE?
3930 012260 003341              BGT     1$              ;NO--BRANCH
3931 012262 000004      EXIT13: SCOPE          ;LOOP
3932
3933 ;*****
3934 ;*TEST 14 TRACK ADDRESSING TEST
3935
3936 ;* THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
3937 ;* IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
3938 ;* GETTING ITS OWN TRACK ADDRESS.
3939 ;* A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO INSURE
3940 ;* THE DATA IS VALID. THEN TRACK 0 IS REWRITTEN AND TRACK 1
3941 ;* THROUGH TRACK 18 IS WRITE CHECKED. THEN TRACK 1 IS
3942 ;* REWRITTEN AND TRACK 2 THROUGH TRACK 18 IS WRITE CHECKED.
3943 ;* THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING TRACK 17
3944 ;* AND WRITE CHECKING TRACK 18.
3945
3946 ;*****
3947 TST14:
3948 012264 033737 001460 001260      BIT      2#BITS+(14*2),TSTNMS ;DO THIS TEST?
3949 012272 001002              BNE     64$            ;YES--BRANCH
3950 012274 000137 012620              JMP     TST15          ;NO--GO TO THE NEXT TEST
3951 012300 013737 002320 001176 64$:  MOV      2#RPT14,$TIMES   ;GET THE ITERATION COUNT
3952 012306 012737 012344 001106      MOV      #TEST14,2#$SLPADR
3953 012314 012737 012264 001110      MOV      #TST14,2#$SLPERR ;SETUP THE ERROR LOOP ADDRESS
3954 012322 012737 000014 001102      MOV      #14,2#$TSTNM    ;SET UP TEST NUMBER AND
3955 ;* CLEAR THE ERROR FLAG (SERFLG)
3956 012330 013737 001102 177570      MOV      $TSTNM,2#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3957 012336 012737 012616 001270      MOV      #EXIT14,2#BYPASS
3958 012344 012706 001100      TEST14: MOV      #STACK,$P    ;SET THE STACK POINTER
3959 012350 004737 025406      JSR      PC,2#FILBUF    ;FILL THE BUFFER WITH TRACK ADDRESS
3960 012354 012737 000161 003556      MOV      #WRITE,2#DTADPB+2 ;COMMAND=WRITE DATA
3961 012362 013737 002322 003566      MOV      2#FC14,2#DTADPB+12 ;CYLINDER
3962 012370 113737 002324 003564      MOVVB   2#FS14,2#DTADPB+10 ;SECTOR
3963 012376 012737 177400 003560      MOV      #SCTRWC,2#DTADPB+4 ;WORD COUNT
3964 012404 012737 044134 003562      MOV      #BUFFER,2#DTADPB+6 ;BUFFER ADDRESS
3965 012412 005000              CLR      RO            ;TRACK=0
3966 012414 110037 003565 1$:  MOVVB   RO,2#DTADPB+11   ;TRACK ADDRESS
3967 012420 004037 023342      JSR      RO,2#DRVCAL     ;START A DATA TRANSFER
3968 012424 062737 001000 003562      ADD      #10256*2,2#DTADPB+6 ;UPDATE BUFFER ADDRESS
3969 012432 005200              INC      RO            ;UPDATE TRACK NUMBER
3970 012434 022700 000023      CMP      #1019,RO       ;OUT OF TRACKS?
  
```

3971	012440	003365			BGT	1\$;NO--BRANCH
3972	012442	012737	044134	003562	MOV	#BUFFER,2#DTADPB+6	;BUFFER ADDRESS
3973	012450	005000			CLR	RO	
3974	012452	012737	000151	003556	MOV	#WRCKD,2#DTADPB+2	;COMMAND=WRITE CHECK
3975	012460	110037	003565	2\$:	MOVB	RO,2#DTADPB+11	;TRACK ADDRESS
3976	012464	004037	023342		JSR	RO,2#DRVCAL	;START A DATA TRANSFER
3977	012470	062737	001000	003562	ADD	#D256*2,2#DTADPB+6	;UPDATE BUFFER ADDRESS
3978	012476	005200			INC	RO	;UPDATE TRACK NUMBER
3979	012500	022700	000023		CMP	#D19,RO	;OUT OF TRACKS?
3980	012504	003365			BGT	2\$;NO--BRANCH
3981	012506	005000			CLR	RO	;FIRST TRACK ADDRESS
3982	012510	110037	003565	3\$:	MOVB	RO,2#DTADPB+11	;TRACK
3983	012514	010001			MOV	RO,R1	;FORM BUFFER ADDRESS
3984	012516	012737	044134	003562	MOV	#BUFFER,2#DTADPB+6	;BUFFER ADDRESS
3985	012524	005301		4\$:	DEC	R1	
3986	012526	002404			BLT	5\$	
3987	012530	062737	001000	003562	ADD	#D256*2,2#DTADPB+6	
3988	012536	000772			BR	4\$	
3989	012540	012737	000161	003556	MOV	#WRITE,2#DTADPB+2	;COMMAND=WRITE DATA
3990	012546	004037	023342		JSR	RO,2#DRVCAL	;START A DATA TRANSFER
3991	012552	062737	001000	003562	ADD	#D256*2,2#DTADPB+6	;UPDATE BUFFER ADDRESS
3992	012560	105237	003565		INCB	2#DTADPB+11	;MOVE TO NEXT TRACK
3993	012564	012737	000151	003556	MOV	#WRCKD,2#DTADPB+2	;COMMAND=WRITE CHECK DATA
3994	012572	004037	023342		JSR	RO,2#DRVCAL	;START A DATA TRANSFER
3995	012576	122737	000022	003565	CMPB	#D18,2#DTADPB+11	;OUT OF TRACKS?
3996	012604	003362			BGT	6\$;NO--BRANCH
3997	012606	005200			INC	RO	;NEXT TRACK TO WRITE
3998	012610	022700	000022		CMP	#D18,RO	;OUT OF TRACKS?
3999	012614	003335			BGT	3\$;NO--BRANCH
4000	012616	000004					

EXIT14: SCOPE



4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056

.SBTTL *** DATA TEST (15) ***

;*TEST 15 DATA TEST

THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS
"FC" THROUGH "LC" BY THE INCREMENT "IC" USING THE DATA PATTERNS
SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
1. SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
2. WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
3. READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
4. INCREMENT "NT" BY "IT"
5. REPEAT STEPS 1-4 FOR EACH DATA PATTERN
6. REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"

IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND
THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS
ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE
WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS
FATAL AND NO READ OCCURS.
FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
PAT DEFAULTS TO 177777 (ALL POSSIBLE PATTERNS)
THE POSSIBLE PATTERNS ARE:

PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
165555	000001	177776	000000	000000	052525	007417	026455
133333	000003	177774	000000	010421	052525	007417	026455
165555	000007	177770	000000	021042	052525	007417	026455
133333	000017	177760	177777	031463	125252	170360	151322
165555	000037	177740	177777	042104	125252	170360	151322
133333	000077	177700	177777	052525	125252	170360	151322
165555	000177	177600	000000	063146	052525	007417	026455
133333	000377	177400	000000	073567	052525	007417	026455
165555	000777	177000	177777	104210	125252	170360	151322
133333	001777	176000	177777	114631	125252	170360	151322
165555	003777	174000	000000	125252	052525	007417	026455
133333	007777	170000	177777	135673	125252	170360	151322
165555	017777	160000	000000	146314	052525	007417	026455
133333	037777	140000	177777	156735	125252	170360	151322
165555	077777	100000	000000	167356	052525	007417	026455
133333	177777	000000	177777	177777	125252	170360	151322
PAT 8	PAT 9	PAT 10	PAT 11	PAT 12	PAT 13	PAT 14	PAT 15
165555	000001	177776	172666	077777	153333	000000	177777
133333	000002	177775	155555	137777	066667	177777	000000
165555	000004	177773	172666	157777	153333	177777	000000
133333	000010	177767	155555	167777	066667	177777	000000
165555	000020	177757	172666	173777	153333	177777	000000
133333	000040	177737	155555	175777	066667	177777	000000
165555	000100	177677	172666	176777	153333	177777	000000
133333	000200	177577	155555	177377	066667	177777	000000
165555	000400	177377	172666	177577	153333	177777	000000
133333	001000	176777	155555	177677	066667	177777	000000
165555	002000	175777	172666	177737	153333	177777	000000

```

4057      ;*      133333 004000 173777 155555 177757 066667 177777 000000
4058      ;*      165555 010000 167777 172666 177767 153333 177777 000000
4059      ;*      133333 020000 157777 155555 177773 066667 177777 000000
4060      ;*      165555 040000 137777 172666 177775 153333 177777 000000
4061      ;*      133333 100000 077777 155555 177776 066667 177777 000000
4062      ;*
4063      ;*
4064      ;*

```

```

4065      012620      ;TST15:
4066      012620 033737 001462 001260      BIT      @#BITS+<15*2>,TSTNMS ;DO THIS TEST?
4067      012626 001002      BNE      64$      ;YES--BRANCH
4068      012630 000137 013260      JMP      TST16      ;NO--GO TO THE NEXT TEST
4069      012634 013737 002330 001176 64$:      MOV      @#RPT15,$TIMES ;GET THE ITERATION COUNT
4070      012642 012737 013004 001106      MOV      @TEST15,@#SLPADR
4071      012650 012737 012620 001110      MOV      @TST15,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
4072      012656 012737 000015 001102      MOV      @15,@#$TSTNM ;SET UP TEST NUMBER AND
4073      ;CLEAR THE ERROR FLAG ($ERFLG)
4074      012664 013737 001102 177570      MOV      $TSTNM,@#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4075      012672 005000      CLR      R0      ;CLEAR SWITCH
4076      012674 005004      CLR      R4      ;FORM WORD COUNT IN R4
4077      012676 013701 002350      MOV      @#LS15,R1
4078      012702 163701 002346      SUB      @#FS15,R1
4079      012706 002003      BGE      1$      ;BRANCH IF FS < OR = LS
4080      012710 062701 000026      ADD      @#D22,R1 ;MAKE DIFFERENCE POSITIVE
4081      012714 005100      COM      R0      ;SET SWITCH
4082      012716 062704 000400 1$:      ADD      @#D256,R4
4083      012722 005301      DEC      R1
4084      012724 002374      BGE      1$
4085      012726 005404      NEG      R4
4086      012730 010405      MOV      R4,R5 ;COPY NORMAL WORD COUNT INTO SMALL WC
4087      012732 005700      TST      R0 ;SWITCH SET?
4088      012734 001412      BEQ      3$      ;NO--BRANCH
4089      012736 005005      CLR      R5 ;FORM WORD COUNT FOR LS < FS
4090      012740 012701 000025      MOV      @#D21,R1
4091      012744 163701 002346      SUB      @#FS15,R1
4092      012750 062705 000400 2$:      ADD      @#D256,R5
4093      012754 005301      DEC      R1
4094      012756 002374      BGE      2$
4095      012760 005405      NEG      R5
4096      012762 113737 002346 003564 3$:      MOV      @#FS15,@#DTADPB+10 ;SECTOR
4097      012770 012737 044134 003562      MOV      @BUFFER,@#DTADPB+6 ;DATA BUFFER
4098      012776 012737 013256 001270      MOV      @EXIT15,@#BYPASS
4099      013004 005037 001352      TEST15: CLR      @#WCEFLG ;CLEAR THE WRITE CHECK ERROR FLAG
4100      013010 013701 002332      MOV      @#FC15,R1 ;PICKUP FIRST CYLINDER
4101      013014 000407      BR      2$
4102      013016 005720 1$:      TST      (R0)+ ;MOVE TO NEXT DATA PATTERN
4103      013020 022700 000040      CMP      @#D16*2,R0 ;OUT OF PATTERNS?
4104      013024 003004      BGT      3$      ;NO--BRANCH
4105      013026 004037 025356      JSR      R0,@#INCCYL ;MOVE TO NEXT CYLINDER
4106      013032 000511      BR      EXIT15 ;OUT OF CYLINDERS
4107      013034 005000 2$:      CLR      R0 ;START WITH PATTERN 0
4108      013036 036037 001430 002352 3$:      BIT      BITS(R0),@#PTRN15 ;THIS PATTERN SELECTED?
4109      013044 001764      BEQ      1$      ;NO--BRANCH
4110      013046 013702 002340      MOV      @#FT15,R2 ;FIRST TRACK
4111      013052 010137 003566      MOV      R1,@#DTADPB+12 ;CYLINDER
4112      013056 110237 003565 4$:      MOV      R2,@#DTADPB+11 ;TRACK

```


4113	013062	010437	003560		MOV	R4, @#DTADPB+4	:WORD COUNT
4114	013066	022701	000632		CMP	#410., R1	:LAST DISK CYLINDER?
4115	013072	003005			BGT	5\$:NO--BRANCH
4116	013074	022702	000022		CMP	#18., R2	:LAST DISK TRACK?
4117	013100	003002			BGT	5\$:NO--BRANCH
4118	013102	010537	003560		MOV	R5, @#DTADPB+4	:SHORT WORD COUNT
4119	013106	013703	177570	5\$:	MOV	@#SWR, R3	:INHIBIT WRITE AND
4120	013112	005103			COM	R3	:WRITE CHECK?
4121	013114	032703	000030		BIT	#SW04!SW03, R3	
4122	013120	001424			BEQ	7\$:YES--BRANCH
4123	013122	004737	026076		JSR	PC, @#SETBUF	:MOVE DATA PATTERN INTO THE BUFFER
4124	013126	032737	000020	177570	BIT	#SW04, @#SWR	:INHIBIT WRITE?
4125	013134	001005			BNE	6\$:YES--BRANCH
4126	013136	012737	000161	003556	MOV	#WRITE, @#DTADPB+2	:COMMAND=WRITE DATA
4127	013144	004037	023342		JSR	RD, @#DRVCAL	:START A DATA TRANSFER
4128	013150	032737	000010	177570	BIT	#SW03, @#SWR	:INHIBIT WRITE CHECK?
4129	013156	001005			BNE	7\$:YES--BRANCH
4130	013160	012737	000151	003556	MOV	#WRCKD, @#DTADPB+2	:COMMAND=WRITE CHECK DATA
4131	013166	004037	023342		JSR	RD, @#DRVCAL	:START A DATA TRANSFER
4132	013172	005737	001352		TST	@#WCEFLG	:WRITE CHECK ERROR FLAG SET?
4133	013176	001404			BEQ	8\$:NO--BRANCH
4134	013200	032737	000001	177570	BIT	#SW00, @#SWR	:PERFORM READ AFTER FATAL "WCE"?
4135	013206	001417			BEQ	9\$:NO--BRANCH
4136	013210	032737	000004	177570	BIT	#SW02, @#SWR	:INHIBIT READ DATA AND SOFTWARE COMPARE?
4137	013216	001013			BNE	9\$:YES--BRANCH
4138	013220	012737	000171	003556	MOV	#READ, @#DTADPB+2	:COMMAND=READ
4139	013226	004037	023342		JSR	RD, @#DRVCAL	:START A DATA TRANSFER
4140	013232	032737	000002	177570	BIT	#SW01, @#SWR	:COMPARE THE DATA?
4141	013240	001002			BNE	9\$:NO--BRANCH
4142	013242	004737	026206		JSR	PC, @#DATCMP	:YES--DO IT
4143	013246	004037	025326	9\$:	JSR	RD, @#INCTRK	:MOVE TO NEXT TRACK
4144	013252	000661			BR	1\$:OUT OF TRACKS GO TO NEXT PATTERN
4145	013254	000700			BR	4\$:LOOP
4146	013256	000004			EXIT15:	SCOPE	:SCOPE LOOP
4147							
4148							
4149							
4150							
4151							
4152							

4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208

.SBTTL *** EXERCISE TEST (16) ***

;*TEST 16 RANDOM ADDRESS AND RANDOM PATTERN TEST (TST16)

;* STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK
;* IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS
;* OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR
;* FOR THAT SECTOR.
;* THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES
;* "R" DEFAULTS TO 20,000.

- 1) GENERATE A RANDOM ADDRESS
- 2) WRITE A RANDOM PATTERN AT THE ADDRESS GENERATED IN 1.
- 3) GENERATE A RANDOM ADDRESS
- 4) READ THE SECTOR AT THE ADDRESS GENERATED IN 3.
- 5) DO A SOFTWARE CHECK OF THE DATA READ IN 4.
- 6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2
- 7) GENERATE A RANDOM ADDRESS
- 8) READ THE SECTOR AT THE ADDRESS GENERATED IN 7.
- 9) DO A SOFTWARE CHECK OF THE DATA READ IN 8
- 10) DO A WRITE CHECK OF THE DATA WRITTEN IN 2

TST16:

```

BIT      @#BITS+<16*2>,TSTNMS ;DO THIS TEST?
BNE      64$                    ;YES--BRANCH
JMP      TST17                  ;NO--GO TO THE NEXT TEST
64$:     MOV      @#RPT16,$TIMES ;GET THE ITERATION COUNT
        MOV      @TST16,@#SLPADR ;SETUP THE SCOPE LOOP ADDRESS
        MOV      @TST16,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
        MOV      @16,@#$TSTNM   ;SET UP TEST NUMBER AND
                                ;CLEAR THE ERROR FLAG ($ERFLG)
                                ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
        MOV      $TSTNM,@#DISPLAY
        MOV      @EXIT16,@#BYPASS
        MOV      @176543,@#$HINUM ;PRIME THE RANDOM NUMBER GENERATOR
        MOV      @123456,@#$LONUM
        MOV      @#FC16,@#DTADPB+12 ;CYLINDER
        MOV      @#TRCKWC,@#DTADPB+4 ;WORD COUNT
        MOV      @#BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
        MOV      @#WRITE,@#DTADPB+2 ;COMMMAND
        BIT      @#SW15,@#C.SWR   ;WRITE THE DISK PACK BEFORE TESTING?
        BNE      3$                    ;NO--BRANCH
        JSR      RO,@#FILRAN      ;FILL DATA BUFFER WITH RANDOM DATA
        CLR      @#DTADPB+10      ;SECTOR AND TRACK
1$:      JSR      RO,@#DRVCAL      ;START A DATA TRANSFER
2$:      INCB    @#DTADPB+11      ;NEXT TRACK
        CMPB    @#D19,@#DTADPB+11 ;TIME FOR NEXT CYLINDER
        BGT     2$                    ;NO--BRANCH

```

1\$:
2\$:

013260				
013260	033737	001464	001260	
013266	001002			
013270	000137	013726		
013274	013737	002356	001176	
013302	012737	013260	001106	
013310	012737	013260	001110	
013316	012737	000016	001102	
013324	013737	001102	177570	
013332	012737	013724	001270	
013340	012737	176543	022022	
013346	012737	123456	022024	
013354	013737	002360	003566	
013362	012737	165000	003560	
013370	012737	044134	003562	
013376	012737	000161	003556	
013404	032737	100000	001246	
013412	001022			
013414	004037	026562		
013420	005037	003564		
013424				
013424	004037	023342		
013430	105237	003565		
013434	122737	000023	003565	
013442	003370			

4209	013444	005237	003566		INC	@#DTADPB+12
4210	013450	023737	002362	003566	CMP	@#LC16,@#DTADPB+12 ;OUT OF CYLINDERS?
4211	013456	002360			BGE	IS ;NO--BRANCH
4212	013460	012737	177400	003560	3\$: MOV	#SCTRWC,@#DTADPB+4 ;WORD COUNT
4213	013466	012737	013502	001106	MOV	#TEST16,@#SLPADR
4214	013474	012737	013502	001110	MOV	#TEST16,@#SLPERR
4215	013502	012706	001100		TEST16: MOV	#STACK,SP ;SET STACK POINTER
4216	013506	004037	027122		JSR	RO,@#RANADR ;GENERATE A RANDOM ADDRESS
4217	013512	013737	003564	001356	MOV	@#DTADPB+10,@#SVADR ;SAVE THE TRACK/SECTOR
4218	013520	013737	003566	001360	MOV	@#DTADPB+12,@#SVADR+2 ;SAVE THE CYLINDER
4219	013526	012737	000161	003556	MOV	#WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
4220	013534	012701	044134		MOV	#BUFFER,R1 ;BUFFER ADDRESS
4221	013540	010137	003562		MOV	R1,@#DTADPB+6
4222	013544	004037	027066		JSR	RO,@#RANPAT ;GENERATE RANDOM PATTERN
4223	013550	004037	023342		JSR	RO,@#DRVCAL ;START A DATA TRANSFER
4224	013554	004037	027122		JSR	RO,@#RANADR
4225	013560	012737	000171	003556	MOV	#READ,@#DTADPB+2 ;COMMAND=READ DATA
4226	013566	012737	045134	003562	MOV	#BUFFER+512,@#DTADPB+6 ;BUFFER ADDRESS
4227	013574	004037	023342		JSR	RO,@#DRVCAL ;START A DATA TRANSFER
4228	013600	004037	026620		JSR	RO,@#RANCK ;CHECK THE DATA
4229	013604	013737	001356	003564	MOV	@#SVADR,@#DTADPB+10 ;GET ADDRESS OF WHERE THE LAST
4230	013612	013737	001360	003566	MOV	@#SVADR+2,@#DTADPB+12 ;WRITE WAS PERFORMED
4231	013620	012737	000151	003556	MOV	#WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
4232	013626	012737	044134	003562	MOV	#BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
4233	013634	004037	023342		JSR	RO,@#DRVCAL ;START A DATA TRANSFER
4234	013640	004037	027122		JSR	RO,@#RANADR ;GENERATE A RANDOM ADDRESS
4235	013644	012737	000171	003556	MOV	#READ,@#DTADPB+2 ;COMMAND=READ
4236	013652	012737	045134	003562	MOV	#BUFFER+512,@#DTADPB+6 ;DATA BUFFER ADDRESS
4237	013660	004037	023342		JSR	RO,@#DRVCAL ;START A DATA TRANSFER
4238	013664	004037	026620		JSR	RO,@#RANCK ;CHECK THE DATA
4239	013670	013737	001356	003564	MOV	@#SVADR,@#DTADPB+10 ;GET DISK ADDRESS OF THE
4240	013676	013737	001360	003566	MOV	@#SVADR+2,@#DTADPB+12 ;LAST WRITE
4241	013704	012737	000151	003556	MOV	#WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
4242	013712	012737	044134	003562	MOV	#BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
4243	013720	004037	023342		JSR	RO,@#DRVCAL ;START A DATA TRANSFER
4244	013724	000004			EXIT16: SCOPE	;LOOP

4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300

.SBTTL *** DATA TEST WITH ALL MEMORY (17) ***

;*TEST 17 DATA TEST WITH ALL MEMORY

;* THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS
;* 409 THROUGH 410 USING THE DATA PATTERNS
;* SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
;* 1. SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
;* 2. WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
;* 3. READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
;* 4. INCREMENT "NT" BY "IT"
;* 5. REPEAT STEPS 1-4 FOR EACH DATA PATTERN
;* 6. REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"

;* IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND
;* THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS
;* ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE
;* WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS
;* FATAL AND NO READ OCCURS.
;* FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
;* PAT DEFAULTS TO 17777 (ALL POSSIBLE PATTERNS)
;* THE POSSIBLE PATTERNS ARE:

PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
165555	000001	177776	000000	000000	052525	007417	026455
133333	000003	177774	000000	010421	052525	007417	026455
165555	000007	177770	000000	021042	052525	007417	026455
133333	000017	177760	177777	031463	125252	170360	151322
165555	000037	177740	177777	042104	125252	170360	151322
133333	000077	177700	177777	052525	125252	170360	151322
165555	000177	177600	000000	063146	052525	007417	026455
133333	000377	177400	000000	073567	052525	007417	026455
165555	000777	177000	177777	104210	125252	170360	151322
133333	001777	176000	177777	114631	125252	170360	151322
165555	003777	174000	000000	125252	052525	007417	026455
133333	007777	170000	177777	135673	125252	170360	151322
165555	017777	160000	000000	146314	052525	007417	026455
133333	037777	140000	177777	156735	125252	170360	151322
165555	077777	100000	000000	167356	052525	007417	026455
133333	177777	000000	177777	177777	125252	170360	151322

;*ST17:

```

BIT      @#BITS+<17*2>,TSTNMS ;DO THIS TEST?
BNE      64$                    ;YES--BRANCH
JMP      $EOP                    ;NO--GO TO THE END OF THE PROGRAM
64$:     MOV      @#RPT17,$TIMES ;GET THE ITERATION COUNT
MOV      @#TEST17,@#SLPADR
MOV      @#TST17,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
MOV      @#17,@#$TSTNM          ;SET UP TEST NUMBER AND
;CLEAR THE ERROR FLAG ($ERFLG)
MOV      $TSTNM,@#DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER

```

```

4301 .....
4302 .....
4303 .....
4304 .....
4305 .....
4306 .....
4307 .....
4308 .....
4309 014000 032737 100000 022064 BIT #100000,$KT11
4310 014006 001002 BNE 1$
4311 014010 000137 014744 JMP SEOP
4312 014014 1$:
4313 014014 005037 172340 CLR KIPAR0
4314 014020 012737 000200 172342 MOV #200,KIPAR1
4315 014026 012737 000400 172344 MOV #400,KIPAR2
4316 014034 012737 000600 172346 MOV #600,KIPAR3
4317 014042 012737 001000 172350 MOV #1000,KIPAR4
4318 014050 012737 001200 172352 MOV #1200,KIPAR5
4319 014056 012737 001400 172354 MOV #1400,KIPAR6
4320 014064 012737 177600 172356 MOV #177600,KIPAR7
4321
4322
4323 014072 012746 077406 MOV #77406,-(SP)
4324 014076 011637 172300 MOV (SP),KIPDR0
4325 014102 011637 172302 MOV (SP),KIPDR1
4326 014106 011637 172304 MOV (SP),KIPDR2
4327 014112 011637 172306 MOV (SP),KIPDR3
4328 014116 011637 172310 MOV (SP),KIPDR4
4329 014122 011637 172312 MOV (SP),KIPDR5
4330 014126 011637 172314 MOV (SP),KIPDR6
4331 014132 011637 172316 MOV (SP),KIPDR7
4332
4333
4334 014136 012737 000020 172516 MOV #20,SR3
4335 014144 012737 000001 177572 MOV #1,SRO
4336 014152 012737 044134 001212 T15.1: MOV #BUFFER,BUF
4337 014160 032737 017776 001212 BIT #17776,BUF
4338 014166 001406 BEQ 4$
4339 014170 062737 020000 001212 ADD #20000,BUF
4340 014176 042737 017776 001212 BIC #17776,BUF
4341 014204 013746 001212 4$: MOV BUF,-(SP)
4342 014210 000316 SWAB (SP)
4343 014212 006216 ASR (SP)
4344 014214 006216 ASR (SP)
4345 014216 006216 ASR (SP)
4346 014220 006216 ASR (SP)
4347 014222 006216 ASR (SP)
4348 014224 005037 001220 CLR TEMP3
4349 014230 005037 172352 CLR KIPAR5
4350 014234 062737 000200 172352 5$: ADD #200,KIPAR5
4351 014242 005237 001220 INC TEMP3
4352 014246 005316 DEC (SP)
4353 014250 001371 BNE 5$
4354 014252 005726 TST (SP)+
4355 014254 013737 17235 172354 MOV KIPAR5,KIPAR6
4356 014262 062737 000200 172354 ADD #200,KIPAR6

```


4357	014270	013737	001214	001216		MOV	TEMP1,TEMP2	
4358	014276	162737	000003	001216		SUB	#3,TEMP2	
4359	014304	012737	120000	001212	T15.2:	MOV	#120000,BUF	
4360	014312	005000				CLR	R0	:CLEAR SWITCH
4361	014314	005004				CLR	R4	:FORM WORD COUNT IN R4
4362	014316	013701	002350			MOV	#LS15,R1	
4363	014322	163701	002346			SUB	#FS15,R1	
4364	014326	002003				BGE	1\$:BRANCH IF FS < OR = LS
4365	014330	062701	000026			ADD	#D22,R1	:MAKE DIFFERENCE POSITIVE
4366	014334	005100				COM	R0	:SET SWITCH
4367	014336	062704	000400		1\$:	ADD	#D256,R4	
4368	014342	005301				DEC	R1	
4369	014344	002374				BGE	1\$	
4370	014346	005404				NEG	R4	
4371	014350	010405				MOV	R4,R5	:COPY NORMAL WORD COUNT INTO SMALL WC
4372	014352	005700				TST	R0	:SWITCH SET?
4373	014354	001412				BEQ	3\$:NO--BRANCH
4374	014356	005005				CLR	R5	:FORM WORD COUNT FOR LS < FS
4375	014360	012701	000025			MOV	#D21,R1	
4376	014364	163701	002346			SUB	#FS15,R1	
4377	014370	062705	000400		2\$:	ADD	#D256,R5	
4378	014374	005301				DEC	R1	
4379	014376	002374				BGE	2\$	
4380	014400	005405				NEG	R5	
4381	014402	113737	002346	003564	3\$:	MOVB	#FS15,#DTADPB+10	:SECTOR
4382	014410	013737	001212	003562		MOV	BUF,#DTADPB+6	:DATA BUFFER
4383	014416	012737	014676	001270		MOV	#EXIT17,#B/PASS	
4384	014424	005037	001352		TEST17:	CLR	#WCEFLG	:CLEAR THE WRITE CHECK ERROR FLAG
4385	014430	012701	000631			MOV	#409.,R1	:PICKUP FIRST CYLINDER
4386	014434	000407				BR	2\$	
4387	014436	005720			1\$:	TST	(R0)+	:MOVE TO NEXT DATA PATTERN
4388	014440	022700	000020			CMP	#D8*2,R0	:OUT OF PATTERNS?
4389	014444	003004				BGT	3\$:NO--BRANCH
4390	014446	004037	025356			JSR	R0,#INCCYL	:MOVE TO NEXT CYLINDER
4391	014452	000511				BR	EXIT17	:OUT OF CYLINDERS
4392	014454	005000			2\$:	CLR	R0	:START WITH PATTERN 0
4393	014456	036037	001430	002352	3\$:	BIT	BITS(R0),#PTRN15	:THIS PATTERN SELECTED?
4394	014464	001764				BEQ	1\$:NO--BRANCH
4395	014466	013702	002340			MOV	#FT15,R2	:FIRST TRACK
4396	014472	010137	003566			MOV	R1,#DTADPB+12	:CYLINDER
4397	014476	110237	003565		4\$:	MOVB	R2,#DTADPB+11	:TRACK
4398	014502	010437	003560			MOV	R4,#DTADPB+4	:WORD COUNT
4399	014506	022701	000632			CMP	#410.,R1	:LAST DISK CYLINDER?
4400	014512	003005				BGT	5\$:NO--BRANCH
4401	014514	022702	000022			CMP	#18.,R2	:LAST DISK TRACK?
4402	014520	003002				BGT	5\$:NO--BRANCH
4403	014522	010537	003560			MOV	R5,#DTADPB+4	:SHORT WORD COUNT
4404	014526	013703	177570		5\$:	MOV	#SWR,R3	:INHIBIT WRITE AND
4405	014532	005103				COM	R3	:WRITE CHECK?
4406	014534	032703	000030			BIT	#SW04!SW03,R3	
4407	014540	001424				BEQ	7\$:YES--BRANCH
4408	014542	004737	026076			JSR	PC,#SETBUF	:MOVE DATA PATTERN INTO THE BUFFER
4409	014546	032737	000020	177570		BIT	#SW04,#SWR	:INHIBIT WRITE?
4410	014554	001005				BNE	6\$:YES--BRANCH
4411	014556	012737	000161	003556		MOV	#WRITE,#DTADPB+2	:COMMAND=WRITE DATA
4412	014564	004037	023342			JSR	R0,#DRVCL	:START A DATA TRANSFER

4413	014570	032737	000010	177570	6\$:	BIT	#SW03, @#SWR	: INHIBIT WRITE CHECK?
4414	014576	001005				BNE	7\$: YES--BRANCH
4415	014600	012737	000151	003556		MOV	#WRCKD, @#DTADPB+2	: COMMAND=WRITE CHECK DATA
4416	014606	004037	023342			JSR	RO, @#DRVCAL	: START A DATA TRANSFER
4417	014612	005737	001352		7\$:	TST	@#WCEFLG	: WRITE CHECK ERROR FLAG SET?
4418	014616	001404				BEQ	8\$: NO--BRANCH
4419	014620	032737	000001	177570		BIT	#SW00, @#SWR	: PERFORM READ AFTER FATAL "WCE"?
4420	014626	001417				BEQ	9\$: NO--BRANCH
4421	014630	032737	000004	177570	8\$:	BIT	#SW02, @#SWR	: INHIBIT READ DATA AND SOFTWARE COMPARE?
4422	014636	001013				BNE	9\$: YES--BRANCH
4423	014640	012737	000171	003556		MOV	#READ, @#DTADPE+2	: COMMAND=READ
4424	014646	004037	023342			JSR	RO, @#DRVCAL	: START A DATA TRANSFER
4425	014652	032737	000002	177570		BIT	#SW01, @#SWR	: COMPARE THE DATA?
4426	014660	001002				BNE	9\$: NO--BRANCH
4427	014662	004737	026206			JSR	PC, @#DATCMP	: YES--DO IT
4428	014666	004037	025326		9\$:	JSR	RO, @#INCTRK	: MOVE TO NEXT TRACK
4429	014672	000661				BR	1\$: OUT OF TRACKS GO TO NEXT PATTERN
4430	014674	000700				BR	4\$: LOOP
4431	014676	023737	001216	001220	EXIT17:	CMP	TEMP2, TEMP3	
4432	014704	002002				BGE	1\$	
4433	014706	000137	014736			JMP	2\$	
4434	014712	005237	001220		1\$:	INC	TEMP3	
4435	014716	062737	000200	172352		ADD	#200, KIPAR5	
4436	014724	062737	000200	172354		ADD	#200, KIPAR6	
4437	014732	000137	014304			JMP	T15.2	
4438	014736				2\$:			
4439	014736	000004				SCOPE		: SCOPE LOOP
4440	014740	005037	177572			CLR	SRO	

```

4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452 014744
4453 014744 104400 014752
4454 014750 000410
4455
4456 014772
4457 014772 005737 001256
4458 014776 001434
4459 015000 104400 015006
4460 015004 000405
4461
4462 015020
4463 015020 013746 001272
4464 015024 104404
4465 015026 002
4466 015027 000
4467 015030 104400 015036
4468 015034 000412
4469
4470 015062
4471 015062 013746 001112
4472 015066 104402
4473 015070 005037 001112
4474 015074 005037 001102
4475 015100 005037 001176
4476 015104 005237 001100
4477 015110 042737 100000 001100
4478 015116 005327
4479 015120 000010
4480 015122 003036
4481 015124 012737
4482 015126 000010
4483 015130 015120
4484 015132 104400 015140
4485 015136 000407
4486
4487 015156
4488 015156 104400 015224
4489 015162 013700 000042
4490 015166 001414
4491 015170 022700 015210
4492 015174 001005
4493 015176 022760 177777 000002
4494 015204 001005
4495 015206 000005
4496 015210 004710

```

.SBTTL END OF PASS ROUTINE

```

;*INCREMENT THE PASS NUMBER ($PASS)
;*INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO RESTART

```

```

$EOP:
      TYPE      64$+4      ;;TYPE ASCIZ STRING
      BR        64$      ;;GET OVER THE ASCIZ
      ;;.ASCIZ      <15><12><12>/END OF PASS/

64$:
      TST      2#DRVSEL    ;ANY DRIVES SELECTED?
      BEQ      1$         ;NO--BRANCH
      TYPE      65$+4      ;;TYPE ASCIZ STRING
      BR        65$      ;;GET OVER THE ASCIZ
      ;;.ASCIZ      / ON DRIVE/

65$:
      MOV      2#CHKDRV,-(SP) ;;SAVE 2#CHKDRV FOR TYPEOUT
      TYPOS
      .BYTE    2          ;;GO TYPE--OCTAL ASCII
      .BYTE    0          ;;TYPE 2 DIGIT(S)
      TYPE      66$+4      ;;SUPPRESS LEADING ZEROS
      BR        66$      ;;TYPE ASCIZ STRING
      ;;.ASCIZ      / ERRORS DETECTED=/

66$:
      MOV      2#$ERTTL,-(SP) ;;SAVE 2#$ERTTL FOR TYPEOUT
      TYPOC
      CLR      2#$ERTTL   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
      CLR      $STNM      ;;ZERO ERROR TOTAL
      CLR      $TIMES     ;;ZERO THE TEST NUMBER
      INC      $PASS      ;;ZERO THE NUMBER OF ITERATIONS
      BIC      #100000,$PASS ;;INCREMENT THE PASS NUMBER
      DEC      (PC)+      ;;DON'T ALLOW A NEG. NUMBER
      SEOPCT: .WORD      8. ;;LOOP?
      BGT      $DOAGN     ;;YES
      MOV      (PC)+,2(PC)+ ;;RESTORE COUNTER

SENDCT: .WORD      8.

64$:
      TYPE      64$+4      ;;TYPE ASCIZ STRING
      BR        64$      ;;GET OVER THE ASCIZ
      ;;.ASCIZ      <15><12>/END OF TEST/

$GET42:
      TYPE      $ENULL    ;TYPE NULL CHARACTER
      MOV      2#42,RO    ;;GET MONITOR ADDRESS
      BEQ      $DOAGN     ;;BRANCH IF NO MONITOR
      CMP      #SENDAD,RO ;;IS MONITOR ACT11?
      BNE      SENDAD     ;;NO--BRANCH
      CMP      #-1,2(RO)  ;;YES--IS THIS THE LAST PASS?
      BNE      $DOAGN     ;;NO--MAKE ANOTHER PASS
      RESET
      SENDAD: JSR      PC,(RO) ;;CLEAR THE WORLD
                        ;;GO TO MONITOR

```



```

4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558

.SBTTL *** SUBROUTINES ***
;*****
.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 TYPERR 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

$ERROR:
4521 015230 032737 000400 177570 BIT #SW08,2#SWR ;SEND ERROR MESSAGE TO TTY?
4522 015230 001411 BEQ 7$ ;YES--BRANCH
4523 015240 005737 001254 TST 2#LPTAVL ;IS THERE A LINE PRINTER AVAILABLE?
4524 015244 001406 BEQ 7$ ;NO--BRANCH
4525 015246 013737 001414 001142 MOV 2#LPS,2#STPS ;YES--SETUP STATUS
4526 015254 013737 001416 001144 MOV 2#LPB,2#STPB ;AND BUFFER REG.'S FOR LINE PRINTER
4527 015262 105237 001103 7$: INCB SERFLG ;SET THE ERROR FLAG
4528 015266 001775 BEQ 7$ ;DON'T LET THE FLAG GO TO ZERO
4529 015270 013737 001102 177570 MOV $STNM,2#DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG
4530 015276 032737 002000 177570 BIT #BIT10,2#SWR ;BELL ON ERROR?
4531 015304 001402 BEQ 1$ ;NO - SKIP
4532 015306 104400 001202 TYPE $BELL ;RING BELL
4533 015312 005237 001112 1$: INC $ERTTL ;COUNT THE NUMBER OF ERRORS
4534 015316 011637 001116 MOV (SP), $ERRPC ;GET ADDRESS OF ERROR INSTRUCTION
4535 015322 162737 000002 001116 SUB #2, $ERRPC
4536 015330 117737 163562 001114 MOV#B 2#ERRPC, $ITEMB ;STRIP AND SAVE THE ERROR ITEM CODE
4537 015336 032737 020000 177570 BIT #BIT13,2#SWR ;SKIP TYPEOUT IF SET
4538 015344 001004 BNE 2$ ;SKIP TYPEOUTS
4539 015346 004737 015432 JSR PC,2#TYPERR ;GO TO USER ERROR ROUTINE
4540 015352 104400 001207 TYPE $CRLF
4541 015356 005737 177570 2$: TST 2#SWR ;HALT ON ERROR
4542 015362 100001 BPL 3$ ;SKIP IF CONTINUE
4543 015364 000000 HALT ;HALT ON ERROR!
4544 015366 032737 001000 177570 3$: BIT #BIT09,2#SWR ;LOOP ON ERROR SWITCH SET?
4545 015374 001402 BEQ 4$ ;BR IF NO
4546 015376 013716 001110 MOV $LPERR, (SP) ;FUDGE RETURN FOR LOOPING
4547 015402 005737 001200 4$: TST $ESCAPE ;CHECK FOR AN ESCAPE ADDRESS
4548 015406 001402 BEQ 5$ ;BR IF NONE
4549 015410 013716 001200 MOV $ESCAPE, (SP) ;FUDGE RETURN ADDRESS FOR ESCAPE
4550 015414 5$:
4551 015414 013737 001410 001142 MOV 2#TPS,2#STPS ;SET STATUS AND BUFFER REG.'S
4552 015422 013737 001412 001144 MOV 2#TPB,2#STPB ;FOR TTY
4553 015430 000002 RTI ;RETURN FROM ERROR CALL

;*****
.SBTTL TYPERR - TYPE ERROR ROUTINE
; *THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
; *WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR

```



```

4559 ;*TABLE" ($ERRTB), AND REPORTS THE APPROPRIATE INFORMATION
4560 ;*CONCERNING THE ERROR.
4561 ;*CALL
4562 ;*
4563 ;* JSR PC, @#TYPERR
4564 ;* RETURN
4565 015432 113737 001102 001170 TYPERR: MOVB @#STSTNM, @#STMPO ;SAVE THE TEST NUMBER
4566 015440 004037 037744 JSR RO, @#SAVR05 ;SAVE RO-R5
4567 015444 162700 000004 SUB #4, RO ;FORM TEST PC
4568 015450 010037 001154 MOV RO, @#$REG0 ;COPY RO-R5 IN $REG0-$REG5
4569 015454 010137 001156 MOV R1, @#$REG1
4570 015460 010237 001160 MOV R2, @#$REG2
4571 015464 010337 001162 MOV R3, @#$REG3
4572 015470 010437 001164 MOV R4, @#$REG4
4573 015474 010537 001166 MOV R5, @#$REG5
4574 015500 113700 001114 MOVB @#$ITEMB, RO ;PICKUP ERROR ITEM NUMBER
4575 015504 010001 MOV RO, R1 ;AND COPY IT INTO R1
4576 015506 005300 DEC RO ;FORM INDEX FOR ERROR TABLE
4577 015510 106300 ASLB RO
4578 015512 106300 ASLB RO
4579 015514 106300 ASLB RO
4580 015516 103002 BCC 1$ ;IS ERROR > 37?
4581 015520 062700 000220 ADD #ITEM41-$ERRTB, RO ;YES--FORM OFFSET
4582 015524 062700 003644 1$: ADD $ERRTB, RO ;FORM ADDRESS
4583 015530 012037 015544 MOV (RO)+, 2$ ;GET ERROR MESSAGE (EM) POINTER
4584 015534 001447 BEQ 7$ ;BRANCH IF THERE ISN'T ONE
4585 015536 104400 001207 TYPE , $CRLF ;"CARRIAGE RETURN - LINE FEED"
4586 015542 104400 TYPE
4587 015544 000000 2$: .WORD 0 ;"EM" POINTER GOES HERE
4588 015546 162701 000041 SUB #41, R1 ;SPECIAL ERROR ITEM NUMBER?
4589 015552 100440 BMI 7$ ;NO--BRANCH
4590 015554 013701 001276 MOV @#SVSTAT, R1 ;GET STATUS/ERROR INDICATOR
4591 015560 106301 ASLB R1 ;STRIP "DONE" BIT (BIT07)
4592 015562 006301 ASL R1 ;STRIP "ERROR" BIT (BIT15)
4593 015564 012702 001622 MOV #STATBL, R2 ;1ST ADDRESS ON STATUS MESSAGE POINTERS
4594 015570 005003 CLR R3 ;CARRIAGE RETURN-LINE FEED SWITCH
4595 015572 104400 015600 TYPE ,.+4 ;TYPE ASCIZ STRING
4596 015576 000402 BR 64$ ;GET OVER THE ASCIZ
4597 ;;.ASCIZ / (/
4598 015604 64$:
4599 015604 012237 015626 3$: MOV (R2)+, 5$ ;MESSAGE POINTER
4600 015610 006301 ASL R1 ;TYPE THIS MESSAGE?
4601 015612 103013 BCC 6$ ;NO--BRANCH
4602 015614 005103 COM R3 ;YES--TYPE A "CR" & "LF"?
4603 015616 001002 BNE 4$ ;NO--BRANCH
4604 015620 104400 001207 TYPE , $CRLF ;YES
4605 015624 104400 TYPE
4606 015626 000000 5$: .WORD 0 ;MESSAGE POINTER GOES HERE
4607 015630 005701 TST R1 ;MORE TO TYPE?
4608 015632 001403 BEQ 6$ ;NO--BRANCH
4609 015634 104400 040612 TYPE MSG.SP ;YES--SPACES
4610 015640 000761 BR 3$ ;LOOP
4611 015642 001360 6$: BNE 3$ ;BRANCH IF NOT FINISHED
4612 015644 104400 015652 TYPE ,.+4 ;TYPE ASCIZ STRING
4613 015650 000401 BR 65$ ;GET OVER THE ASCIZ
4614 ;;.ASCIZ / (/

```

```

4615 015654 122737 000002 001114 65$:
4616 015654 001421 7$: CMPB #2,SITEMB
4617 015662 122737 000010 001114 BEQ 25$
4618 015664 001415 CMPB #10,SITEMB
4619 015672 122737 000017 001114 BEQ 25$
4620 015702 001411 CMPB #17,SITEMB
4621 015704 122737 000042 001114 BEQ 25$
4622 015712 001405 CMPB #42,SITEMB
4623 015714 122737 000043 001114 BEQ 25$
4624 015722 001401 CMPB #43,SITEMB
4625 015724 000436 BEQ 25$
4626 015726 005737 001234 25$: BR 26$
4627 015732 001033 TST TEMP10
4628 015734 010446 BNE 26$
4629 015736 013704 032360 MOV R4,-(SP)
4630 015742 104400 015750 MOV RPADR,R4
4631 015746 000405 TYPE +4 ;;TYPE ASCIZ STRING
4632 015746 000405 BR 66$ ;;GET OVER THE ASCIZ
4633 ;;.ASCIZ <15><12>/RHBAE= /
4634 015762 016446 000050 65$: MOV RHBAE(R4),-(SP)
4635 015766 104402 TYPOC
4636 015770 104400 015776 TYPE +4 ;;TYPE ASCIZ STRING
4637 015774 000406 BR 67$ ;;GET OVER THE ASCIZ
4638 ;;.ASCIZ / RHCS3= /
4639 016012 016446 000052 67$: MOV RHCS3(R4),-(SP)
4640 016016 104402 TYPOC
4641 016020 012604 MOV (SP)+,R4
4642 016022 012037 016036 26$: MOV (R0)+,8$ ;;PICK UP DATA HEADER (DH) POINTER
4643 016026 001404 BEQ 9$ ;;BRANCH IF NONE
4644 016030 104400 001207 TYPE ,SCRLF ;;CARRIAGE RETURN-LINE FEED
4645 016034 104400 TYPE
4646 016036 000000 8$: .WORD 0 ;;"DH" POINTER GOES HERE
4647 016040 012001 9$: MOV (R0)+,R1 ;;PICKUP DATA TABLE (DT) POINTER
4648 016042 001500 BEQ 20$ ;;BRANCH IF NONE
4649 016044 005005 CLR R5 ;;SET INDENT SWITCH
4650 016046 012000 MOV (R0)+,R0 ;;DATA FORMAT (DF) POINTER
4651 016050 012002 MOV (R0)+,R2 ;;NUMBER OF DH'S TO TYPE
4652 016052 001471 BEQ 17$ ;;BRANCH IF DH NUMBER IS 0
4653 016054 005105 COM R5 ;;NO INDENT
4654 016056 104400 001207 TYPE ,SCRLF ;;CARRIAGE RETURN-LINE FEED
4655 016062 112003 10$: MOVB (R0)+,R3 ;;NUMBER OF DATA WORDS TO TYPE
4656 016064 112004 MOVB (R0)+,R4 ;;AND HOW TO TYPE THEM
4657 016066 006004 11$: ROR R4 ;;OCTAL OR DECIMAL?
4658 016070 103433 BCS 12$ ;;DECIMAL--BRANCH
4659 016072 013146 MOV @R1+,-(SP) ;;SAVE @R1+ FOR TYPEOUT
4660 *****
4661 *****
4662 *****
4663 *****
4664 *****
4665 *****
4666 *****
4667 *****
4668 *****
4669 *****
4670 *****

```


4671	016074	122737	000017	001102		CMPB	#17,\$STSTNM	
4672	016102	001024				BNE	21\$	
4673	016104	122737	000013	001114		CMPB	#13,\$ITEMB	
4674	016112	001007				BNE	22\$	
4675	016114	022702	000001			CMP	#1,R2	
4676	016120	001015				BNE	21\$	
4677	016122	022703	000001			CMP	#1,R3	
4678	016126	001012				BNE	21\$	
4679	016130	000407				BR	23\$	
4680	016132	122737	000014	001114	22\$:	CMPB	#14,\$ITEMB	
4681	016140	001005				BNE	21\$	
4682	016142	022703	000001			CMP	#1,R3	
4683	016146	001002				BNE	21\$	
4684	016150	104422			23\$:	TYP22		
4685	016152	000404				BR	13\$	
4686								
4687	016154	104402			21\$:	TYP0C		;GO TYPE--OCTAL ASCII(ALL DIGITS)
4688	016156	000402				BR	13\$	
4689	016160				12\$:			
4690	016160	013146				MOV	2(R1)+,-(SP)	::SAVE 2(R1)+ FOR TYPEOUT
4691	016162	104410				TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
4692	016164	005303			13\$:	DEC	R3	::MORE NUMBERS TO TYPE?
4693	016166	001403				BEQ	14\$::NO--BRANCH
4694	016170	104400	040612			TYPE	MSG.SP	::YES--TYPE SEPERATORS
4695	016174	000734				BR	11\$::LOOP
4696	016176	005302			14\$:	DEC	R2	::MORE DH'S?
4697	016200	003421				BLE	20\$::NO--BRANCH
4698	016202	104400	001207			TYPE	SCRLF	::YES--START A NEW LINE
4699	016206	005105				COM	R5	::INDENT?
4700	016210	001002				BNE	15\$::NO--BRANCH
4701	016212	104400	040612			TYPE	MSG.SP	::YES--TYPE SPACES
4702	016216	012037	016224		15\$:	MOV	(R0)+,16\$::GET NEXT DH
4703	016222	104400				TYPE		::AND TYPE IT
4704	016224	000000			16\$:	WORD	0	::DH POINTER GOES HERE
4705	016226	104400	001207			TYPE	SCRLF	::CARRIAGE RETURN-LINE FEED
4706	016232	005705				TST	R5	::INDENT?
4707	016234	001312				BNE	10\$::NO--BRANCH
4708	016236	104400	040612		17\$:	TYPE	MSG.SP	::YES--TYPE SPACES
4709	016242	000707				BR	10\$::LOOP
4710	016244	004037	037770		20\$:	JSR	RO,2#GETR05	::RESTORE RO-R5
4711	016250	000207				RTS	PC	::RETURN
4712								
4713								
4714								
4715								
4716								
4717								
4718								
4719								
4720								
4721								
4722	016252							
4723	016252	010046				MOV	RO,-(SP)	::PUSH RO ON STACK
4724	016254	010246				MOV	R2,-(SP)	::PUSH R2 ON STACK
4725	016256	010346				MOV	R3,-(SP)	::PUSH R3 ON STACK
4726	016260	016600	000012			MOV	12(SP),RO	

```

4727 016264 010037 001224      MOV      RO,TEMP5
4728 016270 000300      SKAB     RO
4729 016272 042700 177420      BIC      #177420,RO
4730 016276 006200      ASR      RO
4731 016300 006200      ASR      RO
4732 016302 006200      ASR      RO
4733 016304 006200      ASR      RO
4734
4735
4736 016306 016002 172340      MOV      KIPARO(RO),R2
4737 016312 005003      CLR      R3
4738 016314 006302      ASL      R2
4739 016316 006103      ROL      R3
4740 016320 006302      ASL      R2
4741 016322 006103      ROL      R3
4742 016324 006302      ASL      R2
4743 016326 006103      ROL      R3
4744 016330 006302      ASL      R2
4745 016332 006103      ROL      R3
4746 016334 006302      ASL      R2
4747 016336 006103      ROL      R3
4748 016340 006302      ASL      R2
4749 016342 006103      ROL      R3
4750
4751 016344 042737 160000 001224      BIC      #160000,TEMP5
4752 016352 050237 001224      BIS      R2,TEMP5
4753 016356 010337 001226      MOV      R3,TEMP6
4754 016362 012746 001224      MOV      #TEMP5,-(SP)
4755 016366 004737 021050      JSR      PC,$DB20
4756 016372 062716 000003      ADD      #3,(SP)
4757 016376 012637 016404      MOV      (SP)+,1$
4758 016402 104400
4759 016404 000000      1$:      .WORD 0
4760 016406 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
4761 016410 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
4762 016412 012600      MOV      (SP)+,RO      ;;POP STACK INTO RO
4763 016414 016666 000002 000004      MOV      2(SP),4(SP)
4764 016422 012616      MOV      (SP)+,(SP)
4765 016424 000002      RTI
4766
4767
4768
4769
4770
4771
4772
4773      ;*****
4774      ;*      THIS ROUTINE HANDLES TIME OUT TRAPS THROUGH LOC. 4
4775      ;*****
4776
4777      TIEOUT:
4778 016426 104400 016434      TYPE      +4      ;;TYPE ASCIZ STRING
4779 016432 000433      BR      64$      ;;GET OVER THE ASCIZ
4780
4781 016522      ;;.ASCIZ      <15><12>/TIMEOUT OCCURED THROUGH LOCATION 4 FROM PROGRAM PC/
4782 016522 162716 000002      64$:      SUB      #2,(SP)      ;TYPE PC

```



```

4783 016526 104402
4784 016530 104400 016536
4785 016534 000410
4786
4787 016556
4788 016556 104402
4789 016560 013716 001106
4790 016564 000002
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802 016566
4803 016566 104400 016574
4804 016572 000417
4805
4806 016632
4807 016632 104400 016640
4808 016636 000413
4809
4810 016666
4811 016666 162716 000002
4812 016672 104402
4813 016674 104400 016702
4814 016700 000407
4815
4816 016720
4817 016720 104402
4818 016722 104400 016730
4819 016726 000421
4820
4821 016772
4822 016772 017746 162242
4823 016776 104402
4824 017000 104400 017006
4825 017004 000421
4826
4827 017050
4828 017050 017746 162162
4829 017054 104402
4830 017056 104400 017064
4831 017062 000422
4832
4833 017130
4834 017130 017746 162106
4835 017134 104402
4836 017136 013716 001106
4837 017142 000002
4838

```

```

        TYP0C
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            65$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/PSW WAS /
65$:
        TYP0C
        MOV           $LPADR,(SP)  ;RETURN TO TEST
        RTI

;*****
;* THIS ROUTINE HANDLES PARITY ERRORS
;*****
PARITY:
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            64$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/PARITY TRAP THRU VECTOR 114/
64$:
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            65$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/FROM PROGRAM PC /
65$:
        SUB          #2,(SP)      ;TYPE PC
        TYP0C
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            66$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/PSW WAS /
66$:
        TYP0C
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            67$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><1>/HIGH ERROR ADDRESS REGISTER /
67$:
        MOV          @HERADD,-(SP) ;TYPE HIGH ERROR
        TYP0C
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            68$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/LOW ERROR ADDRESS REGISTER /
68$:
        MOV          @LERADD,-(SP) ;TYPE LOW ERROR
        TYP0C
        TYPE          +4          ;;TYPE ASCIZ STRING
        BR            69$          ;;GET OVER THE ASCIZ
        ;;.ASCIZ      <15><12>/MEMORY SYSTEM ERROR REGISTER /
69$:
        MOV          @MEMERR,-(SP)
        TYP0C
        MOV           $LPADR,(SP) ;RETURN TO TEST
        RTI

```

4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894

017144 105737 001151
 017150 100002
 017152 000000
 017154 000407
 017156 010046
 017160 017600 000002
 017164 112046
 017166 001005
 017170 005726
 017172 012600
 017174 062716 000002
 017200 000002
 017202 004737 017234
 017206 123726 001150
 017212 001364
 017214 013746 001146
 017220 105366 000001
 017224 002770
 017226 004737 017234
 017232 000772
 017234 105777 161702
 017240 100375

```

;*****
.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: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
;*CALL:
;*1) USING A TRAP INSTRUCTION
;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;*OR
;* TYPE
;* MESADR
;*
;*2) USING A JSR INSTRUCTION
;* MOV PS,-(SP) ;;PUSH PROCESSOR STATUS WORD ON THE STACK
;* JSR PC,$TYPE ;;CALL TYPE ROUTINE
;* MESADDR ;;FIRST ADDRESS OF MESSAGE

$TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
BPL 1$ ;;BR IF YES
HALT ;;HALT HERE IF NO TERMINAL
BR 3$ ;;LEAVE
1$: MOV RO,-(SP) ;;SAVE RO
MOV @2(SP),RO ;;GET ADDRESS OF ASCIZ STRING
2$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
MOV (SP)+,RO ;;RESTORE RO
3$: ADD #2,(SP) ;;ADJUST RETURN PC
RTI ;;RETURN
4$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
5$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
BNE 2$ ;;IF NO GO GET NEXT CHAR.
MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
AND THE NULL CHAR.
6$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
BLT 5$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
JSR PC,$TYPEC ;;GO TYPE A NULL
BR 6$ ;;LOOP
$TYPEC: TSTB @2$TPS ;;WAIT UNTIL PRINTER IS READY
BPL $TYPEC
  
```


4895 017242 116677 000002 161674
 4896 017250 000207
 4897
 4898
 4899
 4900
 4901
 4902
 4903
 4904
 4905
 4906
 4907
 4908
 4909
 4910
 4911
 4912
 4913
 4914
 4915
 4916
 4917
 4918
 4919
 4920
 4921
 4922
 4923 017252 017646 000000
 4924 017256 116637 000001 017475
 4925 017264 112637 017477
 4926 017270 062716 000002
 4927 017274 000406
 4928 017276 112737 000001 017475
 4929 017304 112737 000006 017477
 4930 017312 112737 000005 017474
 4931 017320 010346
 4932 017322 010446
 4933 017324 010546
 4934 017326 113704 017477
 4935 017332 005404
 4936 017334 062704 000006
 4937 017340 110437 017476
 4938 017344 113704 017475
 4939 017350 016605 000012
 4940 017354 005003
 4941 017356 006105
 4942 017360 000404
 4943 017362 006105
 4944 017364 006105
 4945 017366 006105
 4946 017370 010503
 4947 017372 006103
 4948 017374 105337 017476
 4949 017400 100016
 4950 017402 042703 177770

```

MOV B 2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
RTS PC
;*****
.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
$TYPOS: MOV @2(SP),-(SP) ;;PICKUP THE MODE
MOV B 1(SP),@OFILL ;;LOAD ZERO FILL SWITCH
MOV B (SP)+,@MODE+1 ;;NUMBER OF DIGITS TO TYPE
ADD #2,(SP) ;;ADJUST RETURN ADDRESS
BR $TYPON
$TYPOC: MOV B #1,@OFILL ;;SET THE ZERO FILL SWITCH
MOV B #6,@MODE+1 ;;SET FOR SIX(6) DIGITS
$TYPON: MOV B #5,@CNT ;;SET THE ITERATION COUNT
MOV R3,-(SP) ;;SAVE R3
MOV R4,-(SP) ;;SAVE R4
MOV R5,-(SP) ;;SAVE R5
MOV B @MODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
NEG R4
ADD #6,R4 ;;SUBTRACT IT FOR MAX. ALLOWED
MOV B R4,@MODE ;;SAVE IT FOR USE
MOV B @OFILL,R4 ;;GET THE ZERO FILL SWITCH
MOV 12(SP),R5 ;;PICKUP THE INPUT NUMBER
CLR R3 ;;CLEAR THE OUTPUT WORD
1$: 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 @MODE ;;TYPE THIS DIGIT?
BPL 7$ ;;BR IF NO
BIC #177770,R3 ;;GET RID OF JUNK

```

```

4951 017406 001002      BNE      4$      ;; TEST FOR 0
4952 017410 005704      TST      R4      ;; SUPPRESS THIS 0?
4953 017412 001403      BEQ      5$      ;; BR IF YES
4954 017414 005204      4$: INC      R4      ;; DON'T SUPPRESS ANYMORE 0'S
4955 017416 052703 000060  BIS      #'0,R3  ;; MAKE THIS DIGIT ASCII
4956 017422 052703 000040  5$: BIS      #' ,R3  ;; MAKE ASCII IF NOT ALREADY
4957 017426 110337 017472  MOV      R3,8$   ;; SAVE FOR TYPING
4958 017432 104400 017472  TYPE     8$     ;; GO TYPE THIS DIGIT
4959 017436 105337 017474  7$: DECB   $OCNT  ;; COUNT BY 1
4960 017442 003347      BGT      2$     ;; BR IF MORE TO DO
4961 017444 002402      BLT      6$     ;; BR IF DONE
4962 017446 005204      INC      R4     ;; INSURE LAST DIGIT ISN'T A BLANK
4963 017450 000744      BR       2$     ;; GO DO THE LAST DIGIT
4964 017452 012605      6$: MOV     (SP)+,R5  ;; RESTORE R5
4965 017454 012604      MOV     (SP)+,R4  ;; RESTORE R4
4966 017456 012603      MOV     (SP)+,R3  ;; RESTORE R3
4967 017460 016666 000002 000004  MOV     2(SP),4(SP) ;; SET THE STACK FOR RETURNING
4968 017466 012616      MOV     (SP)+,(SP)
4969 017470 000002      RTI     ;; RETURN
4970 017472      000      8$: .BYTE   0      ;; STORAGE FOR ASCII DIGIT
4971 017473      000      .BYTE   0      ;; TERMINATOR FOR TYPE ROUTINE
4972 017474      000      $OCNT: .BYTE  0      ;; OCTAL DIGIT COUNTER
4973 017475      000      $OFILL: .BYTE 0      ;; ZERO FILL SWITCH
4974 017476 000000      $OMODE: .WORD 0     ;; NUMBER OF DIGITS TO TYPE
*****
.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
$TYPDS:
MOV     R0,-(SP)      ;; PUSH R0 ON STACK
MOV     R1,-(SP)      ;; PUSH R1 ON STACK
MOV     R2,-(SP)      ;; PUSH R2 ON STACK
MOV     R3,-(SP)      ;; PUSH R3 ON STACK
MOV     R5,-(SP)      ;; PUSH R5 ON STACK
MOV     #20200,-(SP)  ;; SET BLANK SWITCH AND SIGN
MOV     20(SP),R5     ;; GET THE INPUT NUMBER
BPL     1$           ;; BR IF INPUT IS POS.
NEG     R5           ;; MAKE THE BINARY NUMBER POS.
MOVB    #'-,1(SP)    ;; MAKE THE ASCII NUMBER NEG.
1$: CLR     R0       ;; ZERO THE CONSTANTS INDEX
MOV     #$DBLK,R3    ;; SETUP THE OUTPUT POINTER
MOVB    #' ,(R3)+    ;; SET THE FIRST CHARACTER TO A BLANK
2$: CLR     R2       ;; CLEAR THE BCD NUMBER
MOV     $DTBL(R0),R1 ;; GET THE CONSTANT
3$: SUB    R1,R5     ;; FORM THIS BCD DIGIT
BLT     4$         ;; BR IF DONE
INC     R2         ;; INCREASE THE BCD DIGIT BY 1

```



```

5007 017562 000774          BR      3$
5008 017564 060105          4$: ADD   R1,R5          ;; ADD BACK THE CONSTANT
5009 017566 005702          TST   R2              ;; CHECK IF BCD DIGIT=0
5010 017570 001002          BNE   5$              ;; FALL THROUGH IF 0
5011 017572 105716          TSTB  (SP)            ;; STILL DOING LEADING 0'S?
5012 017574 100407          BMI   7$              ;; BR IF YES
5013 017576 106316          5$: ASLB (SP)          ;; MSD?
5014 017600 103003          BCC   6$              ;; BR IF NO
5015 017602 116663 000001 177777 MOVB  1(SP),-1(R3)    ;; YES--SET THE SIGN
5016 017610 052702 000060          6$: BIS   #'0,R2      ;; MAKE THE BCD DIGIT ASCII
5017 017614 052702 000040          7$: BIS   #' ,R2      ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
5018 017620 110223          MOVB  R2,(R3)+        ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
5019 017622 005720          TST  (R0)+            ;; JUST INCREMENTING
5020 017624 020027 000010          CMP   R0,#10         ;; CHECK THE TABLE INDEX
5021 017630 002746          BLT   2$              ;; GO DO THE NEXT DIGIT
5022 017632 003002          BGT   8$              ;; GO TO EXIT
5023 017634 010502          MOV   R5,R2          ;; GET THE LSD
5024 017636 000764          BR    6$              ;; GO CHANGE TO ASCII
5025 017640 105726          8$: TSTB (SP)+        ;; WAS THE LSD THE FIRST NON-ZERO?
5026 017642 100003          BPL   9$              ;; BR IF NO
5027 017644 116663 177777 177776 MOVB  -1(SP),-2(R3)    ;; YES--SET THE SIGN FOR TYPING
5028 017652 105013          9$: CLRB (R3)          ;; SET THE TERMINATOR
5029 017654 012605          MOV  (SP)+,R5        ;; POP STACK INTO R5
5030 017656 012603          MOV  (SP)+,R3        ;; POP STACK INTO R3
5031 017660 012602          MOV  (SP)+,R2        ;; POP STACK INTO R2
5032 017662 012601          MOV  (SP)+,R1        ;; POP STACK INTO R1
5033 017664 012600          MOV  (SP)+,R0        ;; POP STACK INTO R0
5034 017666 104400 017714          TYPE  $DBLK          ;; NOW TYPE THE NUMBER
5035 017672 016666 000002 000004 MOV   2(SP),4(SP)    ;; ADJUST THE STACK
5036 017700 012616          MOV  (SP)+,(SP)
5037 017702 000002          RTI
5038 017704 023420          SDBLK: 10000.
5039 017706 001750          1000.
5040 017710 000144          100.
5041 017712 000012          10.
5042 017714 000004          SDBLK: .BLKW 4
5043          ;*****
5044          .SBTTL TTY INPUT ROUTINE
5045
5046
5047 017724 000000          STKCNT: .WORD 0      ;; NUMBER OF ITEMS IN QUEUE
5048 017726 000000          STKQIN: .WORD 0     ;; INPUT POINTER
5049 017730 000000          STKGOUT: .WORD 0    ;; OUTPUT POINTER
5050 017732 000002          STKQSRT: .BLKB 2    ;; TTY KEYBOARD QUEUE
5051          STKGEND=.
5052
5053          ;*TK INITIALIZE ROUTINE
5054          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
5055          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
5056
5057          ;*CALL
5058          ;* JSR PC,STKINT
5059          ;* RETURN
5060
5061 017734 005037 017724          STKINT: CLR  STKCNT   ;; CLEAR COUNT OF ITEMS IN QUEUE
5062 017740 012737 017732 017726 MOV  #STKQSRT,STKQIN ;; MOVE THE STARTING ADDRESS OF THE

```



```

5063 017746 013737 017726 017730      MOV   $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
5064 017754 012737 020004 000060      MOV   $TKSRV,$TKVEC  ;;INITIALIZE THE KEYBOARD VECTOR
5065 017762 012737 000200 000062      MOV   $200,$TKVEC+2  ;;"BR" LEVEL 4
5066 017770 005777 161144                TST   $TKB           ;;CLEAR DONE FLAG
5067 017774 012777 000100 161134      MOV   $BIT06,$TKS    ;;ENABLE INTERRUPT
5068 020002 000207                RTS                    ;;RETURN TO CALLER
5069
5070
5071      ;*TK SERVICE ROUTINE
5072      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
5073      $TKSRV: MOV    $TKB,-(SP)          ;;PICKUP THE CHARACTER
5074      BIC    $1C177,(SP)              ;;STRIP THE JUNK
5075      CMP    (SP),#3                  ;;IS IT A CONTROL C?
5076      BNE    1$                       ;;BRANCH IF NO
5077      TYPE   $CNTLC                   ;;TYPE A CONTROL-C (1C)
5078      JSR    PC,$TKINT                ;;INIT THE KEYBOARD
5079      JMP    START2                   ;;CONTROL C RESTART
5080 020036 022737 000002 017724 1$:    CMP    $2,$TKCNT          ;;IS THE QUEUE FULL?
5081 020044 001004                BNE    2$                       ;;BRANCH IF NO
5082 020046 104400 001202                TYPE   $BELL              ;;RING THE TTY BELL
5083 020052 005726                TST   (SP)+              ;;CLEAN CHARACTER OFF OF STACK
5084 020054 000002                RTI                    ;;RETURN
5085 020056 005237 017724 2$:    INC    $TKCNT            ;;COUNT THIS CHARACTER
5086 020062 112677 177640                MOV    (SP)+,$TKQIN      ;;AND PUT IT IN QUEUE
5087 020066 005237 017726                INC    $TKQIN           ;;UPDATE THE POINTER
5088 020072 023727 017726 017734      CMP    $TKQIN,$TKQEND   ;;GO OFF THE END?
5089 020100 001003                BNE    3$                       ;;BRANCH IF NO
5090 020102 012737 017732 017726      MOV    $TKQSRRT,$TKQIN  ;;RESET THE POINTER
5091 020110 000002                RTI                    ;;RETURN
5092      ;*****
5093      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
5094      ;*CALL:
5095      ;*   RDCHR                       ;;INPUT A SINGLE CHARACTER FROM THE TTY
5096      ;*   RETURN HERE                 ;;CHARACTER IS ON THE STACK
5097
5098
5099 020112 011646 000004 000002 $RDCHR: MOV    (SP),-(SP)      ;;PUSH DOWN THE PC AND
5100 020114 016666 000004                MOV    4(SP),2(SP)      ;;THE PS
5101 020122 005066 000004                CLR    4(SP)           ;;GET READY FOR A CHARACTER
5102 020126 005037 177776                CLR    $PS            ;;ALLOW INTERRUPTS
5103 020132 005737 017724 1$:    TST   $TKCNT          ;;WAIT ON A CHARACTER
5104 020136 001775                BEQ    1$
5105 020140 005337 017724                DEC    $TKCNT          ;;DECREMENT THE COUNTER
5106 020144 117766 177560 000004      MOV    $TKQOUT,4(SP)   ;;GET ONE CHARACTER
5107 020152 005237 017730                INC    $TKQOUT        ;;UPDATE THE POINTER
5108 020156 023727 017730 017734      CMP    $TKQOUT,$TKQEND ;;DID IT GO OFF OF THE END?
5109 020164 001003                BNE    2$                       ;;BRANCH IF NO
5110 020166 012737 017732 017730      MOV    $TKQSRRT,$TKQOUT ;;RESET THE POINTER
5111 020174 000002                RTI                    ;;RETURN
5112      ;*****
5113      ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
5114      ;*CALL:
5115      ;*   RDLIN                       ;;INPUT A STRING FROM THE TTY
5116      ;*   RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
5117      ;*                               ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
5118

```



```

5119 020176 010346          SRDLIN: MOV      R3, -(SP)          ;; SAVE R3
5120 020200 005046          CLR      -(SP)          ;; CLEAR THE RUBOUT KEY
5121 020202 012703 020437  1$:      MOV      #STTYIN, R3      ;; GET ADDRESS
5122 020206 022703 020463  2$:      CMP      #STTYIN+20., R3      ;; BUFFER FULL?
5123 020212 101456          BLOS     4$              ;; BR IF YES
5124 020214 104412          RDCHR           ;; GO READ ONE CHARACTER FROM THE TTY
5125 020216 112613          MOVVB    (SP)+, (R3)      ;; GET CHARACTER
5126 020220 122713 000177  CMPB     #177, (R3)      ;; IS IT A RUBOUT
5127 020224 001022          BNE     5$              ;; BR IF NO
5128 020226 005716          TST     (SP)           ;; IS THIS THE FIRST RUBOUT?
5129 020230 001007          BNE     6$              ;; BR IF NO
5130 020232 112737 000134 020430  MOVVB    #' \, 9$        ;; TYPE A BACK SLASH
5131 020240 104400 020430  TYPE     9$
5132 020244 012716 177777  MOV      #-1, (SP)      ;; SET THE RUBOUT KEY
5133 020250 005303          DEC     R3              ;; BACKUP BY ONE
5134 020252 020327 020437  6$:      CMP      R3, #STTYIN    ;; STACK EMPTY?
5135 020256 103434          BLO     4$              ;; BR IF YES
5136 020260 111337 020430  MOVVB    (R3), 9$        ;; SETUP TO TYPEOUT THE DELETED CHAR.
5137 020264 104400 020430  TYPE     9$              ;; GO TYPE
5138 020270 000746          BR      2$              ;; GO READ ANOTHER CHAR.
5139 020272 005716          5$:      TST     (SP)           ;; RUBOUT KEY SET?
5140 020274 001406          BEQ     7$              ;; BR IF NO
5141 020276 112737 000134 020430  MOVVB    #' \, 9$        ;; TYPE A BACK SLASH
5142 020304 104400 020430  TYPE     9$
5143 020310 005016          CLR     (SP)           ;; CLEAR THE RUBOUT KEY
5144 020312 122713 000025  7$:      CMPB     #25, (R3)      ;; IS CHARACTER A CTRL U?
5145 020316 001003          BNE     8$              ;; BR IF NO
5146 020320 104400 020432  TYPE     ,SCNTLU        ;; TYPE A CONTROL "U"
5147 020324 000726          BR      1$              ;; GO START OVER
5148 020326 122713 000012  6$:      CMPB     #12, (R3)      ;; IS CHARACTER A "LF"?
5149 020332 001011          BNE     3$              ;; BRANCH IF NO
5150 020334 105013          CLRB   (R3)            ;; CLEAR THE CHARACTER
5151 020336 104400 001207  TYPE     ,SCRLF         ;; TYPE A "CR" & "LF"
5152 020342 104400 020437  TYPE     ,STTYIN        ;; TYPE THE INPUT STRING
5153 020346 000717          BR      2$              ;; GO PICKUP ANOTHER CHARACTER
5154 020350 104400 001206  4$:      TYPE     ,SQUES        ;; TYPE A '?'
5155 020354 000712          BR      1$              ;; CLEAR THE BUFFER AND LOOP
5156 020356 111337 020430  3$:      MOVVB    (R3), 9$        ;; ECHO THE CHARACTER
5157 020362 104400 020430  TYPE     9$
5158 020366 122723 000015  CMPB     #15, (R3)+     ;; CHECK FOR RETURN
5159 020372 001305          BNE     2$              ;; LOOP IF NOT RETURN
5160 020374 105063 177777  CLRB   -1(R3)          ;; CLEAR RETURN (THE 15)
5161 020400 104400 001210  TYPE     ,SLF           ;; TYPE A LINE FEED
5162 020404 005726          TST     (SP)+          ;; CLEAN RUBOUT KEY FROM THE STACK
5163 020406 012603          MOV     (SP)+, R3      ;; RESTORE R3
5164 020410 011646          MOV     (SP), -(SP)     ;; ADJUST THE STACK AND PUT ADDRESS OF THE
5165 020412 016666 000004 000002  MOV     4(SP), 2(SP)    ;; FIRST ASCII CHARACTER ON IT
5166 020420 012766 020437 000004  MOV     #STTYIN, 4(SP)
5167 020426 000002          RTI                    ;; RETURN
5168 020430 000          9$:      .BYTE   0              ;; STORAGE FOR ASCII CHAR. TO TYPE
5169 020431 000          .BYTE   0              ;; TERMINATOR
5170 020432 052536 005015 000  SCNTLU: .ASCIZ /↑U/<15><12> ;; CONTROL "U"
5171 020437 000024          STTYIN: .BLKB 20.      ;; RESERVE 20. BYTES FOR TTY INPUT
5172 020463 136 006503 000012  SCNTLC: .ASCIZ /↑C/<15><12> ;; CONTROL "C"
5173 ;*****
5174

```

```

5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187 020470
5188 020470 006137 177570
5189 020474 100476
5190
5191 020476 000416
5192
5193 020500 013746 000004
5194 020504 012737 020524 000004
5195 020512 005737 177060
5196 020516 012637 000004
5197 020522 000450
5198 020524 022626
5199 020526 012637 000004
5200 020532 000413
5201 020534
5202 020534 105737 001103
5203 020540 001421
5204 020542 123737 001115 001103
5205 020550 101015
5206 020552 032737 001000 177570
5207 020560 001404
5208 020562 013737 001110 001106
5209 020570 000440
5210 020572 105037 001103
5211 020576 005037 001176
5212 020602 000412
5213 020604 032737 004000 177570
5214 020612 001006
5215 020614 005237 001104
5216 020620 023737 001176 001104
5217 020626 002021
5218 020630 012737 000001 001104
5219 020636 013737 020706 001176
5220 020644 105237 001102
5221 020650 011637 001106
5222 020654 011637 001110
5223 020660 005037 001200
5224 020664 112737 000001 001115
5225 020672 013737 001102 177570
5226 020700 013716 001106
5227 020704 000002
5228 020706 000001
5229
5230

```

.SBTTL SCOPE HANDLER ROUTINE

```

; *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
; *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
; *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
; *SW14=1 LOOP ON TEST
; *SW11=1 INHIBIT ITERATIONS
; *SW09=1 LOOP ON ERROR
; *CALL
; * SCOPE ;;SCOPE=IOT

```

```

$SCOPE:
ROL @#SWR ;;LOOP ON PRESENT TEST?
BMI $OVER ;;YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR @#SWR ;;IF RUNNING ON THE "XOR" TESTER CHANGE
;;THIS INSTRUCTION TO A "NOP" (NOP=240)
MOV @#ERRVEC, -(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
MOV @#ERRVEC ;;SET FOR TIMEOUT
TST @#177060 ;;TIME OUT ON XOR?
MOV (SP)+, @#ERRVEC ;;RESTORE THE ERROR VECTOR
BR $SVLAD ;;GO TO THE NEXT TEST
SS: CMP (SP)+, (SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+, @#ERRVEC ;;RESTORE THE ERROR VECTOR
BR 7$ ;;LOOP ON THE PRESENT TEST
6$; *****END OF CODE FOR THE XOR TESTER*****
2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
BEQ 3$ ;;BR IF NO
CMPB $ERMAX, $ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
BHI 3$ ;;BR IF NO
BIT #BIT09, @#SWR ;;LOOP ON ERROR?
BEQ 4$ ;;BR IF NO
7$: MOV $LPERR, $LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
BR $OVER
4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
BR 1$ ;;ESCAPE TO THE NEXT TEST
3$: BIT #BIT11, @#SWR ;;INHIBIT ITERATIONS?
BNE 1$ ;;BR IF YES
INC $ICNT ;;INCREMENT ITERATION COUNT
CMP $TIMES, $ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
BGE $OVER ;;BR IF MORE ITERATION REQUIRED
1$: MOV #1, $ICNT ;;REINITIALIZE THE ITERATION COUNTER
MOV $MXCNT, $TIMES ;;SET NUMBER OF ITERATIONS TO DO
$SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS
MOV (SP), $LPADR ;;SAVE SCOPE LOOP ADDRESS
MOV (SP), $LPERR ;;SAVE ERROR LOOP ADDRESS
CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
MOVB #1, $ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
$OVER: MOV $TSTNM, @#DISPLAY ;;DISPLAY TEST NUMBER
MOV $LPADR, (SP) ;;FUDGE RETURN ADDRESS
RTI ;;FIXES PS
$MXCNT: 1 ;;MAX. NUMBER OF ITERATIONS
;*****

```



```

5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247 020710
5248 020710 010046
5249 020712 010146
5250 020714 010246
5251 020716 010346
5252 020720 010446
5253 020722 010546
5254 020724 016646 000022
5255 020730 016646 000022
5256 020734 016646 000022
5257 020740 016646 000022
5258 020744 000002
5259
5260
5261
5262
5263 020746
5264 020746 012666 000022
5265 020752 012666 000022
5266 020756 012666 000022
5267 020762 012666 000022
5268 020766 012605
5269 020770 012604
5270 020772 012603
5271 020774 012602
5272 020776 012601
5273 021000 012600
5274 021002 000002
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284 021004 010046
5285 021006 016600 000002
5286 021012 005740

```

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES

```

;*SAVE RO-R5
;*CALL:
;* SAVREG
;*UPON RETURN FROM $$SAVREG THE STACK WILL LOOK LIKE:
;*
;*TOP---(+16)
;* +2---(+18)
;* +4---R5
;* +6---R4
;* +8---R3
;*+10---R2
;*+12---R1
;*+14---R0

```

\$\$SAVREG:

```

MOV RO,-(SP) ;;PUSH RO 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 22(SP),-(SP) ;;SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ;;SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ;;SAVE PS OF CALL
MOV 22(SP),-(SP) ;;SAVE PC OF CALL
RTI

```

*RESTORE RO-R5

*CALL:

* RESREG

\$\$RESREG:

```

MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI

```

.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.

```

```

STRAP: MOV RO,-(SP) ;;SAVE RO
MOV 2(SP),RO ;;GET TRAP ADDRESS
TST -(RO) ;;BACKUP BY 2

```

```

5287 021014 111000          MOVB   (R0),R0      ;;GET RIGHT BYTE OF TRAP
5288 021016 016000 021024  MOV    $TRPAD(R0),R0  ;;INDEX TO TABLE
5289 021022 000200          RTS     RO         ;;GO TO ROUTINE

```

.SBTTL TRAP TABLE

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

ROUTINE

\$TRPAD:

```

5299 021024          $TYPE   ;;CALL=TYPE      TRAP+0(104400) TTY TYPEOUT ROUTINE
5300 021024 017144  $TYPOC  ;;CALL=TYPOC   TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
5301 021026 017276  $TYPOS  ;;CALL=TYPOS   TRAP+4(104404) TYPE OCTAL NUMBER (NO LEADING ZEROS)
5302 021030 017252  $TYPON  ;;CALL=TYPON   TRAP+6(104406) TYPE OCTAL NUMBER (AS PER LAST CALL)
5303 021032 017312  $TYPDS  ;;CALL=TYPDS   TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
5304 021034 017500  $RDCHR  ;;CALL=RDCHR   TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
5305 021036 020112  $RDLIN  ;;CALL=RDLIN   TRAP+14(104414) TTY TYPEIN STRING ROUTINE
5306 021040 020176  $SAVREG ;;CALL=SAVREG  TRAP+16(104416) SAVE R0-R5 ROUTINE
5307 021042 020710  $RESREG ;;CALL=RESREG  TRAP+20(104420) RESTORE R0-R5 ROUTINE
5308 021044 020746  TYPE22  ;;CALL=TYP22   TRAP+22(104422) TYPE 22 BITS
5309 021046 016252

```

.SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

;;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
;;*UNSIGNED OCTAL ASCII NUMBER.

CALL

```

5317 *   MOV    #PNTR, -(SP)  ;; POINTER TO LOW WORD OF BINARY NUMBER
5318 *   JSR    PC, @#$DB20  ;; CALL THE ROUTINE
5319 *   RETURN                ;; THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK

```

\$DB20:

```

5322 021050 104416  SAVREG  ;;SAVE ALL REGISTERS
5323 021052 016601 000002  MOV    2(SP),R1  ;;PICKUP THE POINTER TO LOW WORD
5324 021056 012705 021167  MOV    #SOCTVL+13.,R5  ;;POINTER TO DATA TABLE
5325 021062 012704 000014  MOV    #12.,R4  ;;DO ELEVEN CHARACTERS
5326 021066 012703 177770  MOV    #1C7,R3  ;;MASK
5327 021072 012100  MOV    (R1)+,R0  ;;LOWER WORD
5328 021074 012101  MOV    (R1)+,R1  ;;HIGH WORD
5329 021076 005002  CLR    R2  ;;TERMINATOR
5330 021100 110245 15:  MOVB  R2, -(R5)  ;;PUT CHARACTER IN DATA TABLE
5331 021102 010002  MOV    R0,R2  ;;GET THIS DIGIT
5332 021104 005304  DEC    R4  ;;COUNT THIS CHARACTER
5333 021106 003007  BGT   3$  ;;BR IF NOT THE LAST DIGIT
5334 021110 001405  BEQ   2$  ;;BR IF IT IS THE LAST DIGIT
5335 021112 005205  INC   R5  ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
5336 021114 010566 000002  MOV   R5,2(SP)  ;;ASCII CHAR. & PUT IT ON THE STACK
5337 021120 104420  RESREG  ;;RESTORE ALL REGISTERS
5338 021122 000207  RTS    PC  ;;RETURN TO USER
5339 021124 006203 2$:  ASR   R3  ;;POSITION THE MASK FOR THE LAST DIGIT
5340 021126 006001 3$:  ROR  R1  ;;POSITION THE BINARY NUMBER FOR
5341 021130 006000  ROR  R0  ;;
5342 021132 006001  ROR  R1  ;; THE NEXT OCTAL DIGIT

```



```

5399 021300 005502          AJC      R2
5400 021302 062402          ADD     (R4)+,R2
5401 021304 022525          CMP     (R5)+,(R5)+      ;; MOVE TO NEXT TEN POWER
5402 021306 052703 000060  BIS     #'0,R3          ;; CHANGE PARTIAL TO ASCII
5403 021312 110320          MOVVB  R3,(R0)+         ;; SAVE IT
5404 021314 005327          DEC     (PC)+          ;; DONE?
5405 021316 000000          4$:    .WORD 0
5406 021320 001357          BNE    1$              ;; BR IF NO
5407 021322 105020          CLRB   (R0)+          ;; TERMINATOR
5408 021324 104420          RESREG                ;; RESTORE REGISTERS
5409 021326 000207          RTS     PC            ;; RETURN
5410 021330 145000          $TNPWR: 145000         ;; 1.0E09
5411 021332 035632          35632
5412 021334 160400          160400                ;; 1.0E08
5413 021336 002765          2765
5414 021340 113200          113200                ;; 1.0E07
5415 021342 000230          230
5416 021344 041100          041100                ;; 1.0E06
5417 021346 000017          17
5418 021350 103240          103240                ;; 1.0E05
5419 021352 000001          1
5420 021354 023420          23420                 ;; 1.0E04
5421 021356 000000          0
5422 021360 001750          1750                  ;; 1.0E03
5423 021362 000000          0
5424 021364 000144          144                   ;; 1.0E02
5425 021366 000000          0
5426 021370 000012          12                    ;; 1.0E01
5427 021372 000000          0
5428 021374 000001          1                      ;; 1.0E00
5429 021376 000000          0
5430 021400 000014          $DECVL: .BLKB 12.     ;; RESERVE STORAGE FOR ASCIZ STRING
5431                                     ;*****
5432                                     .SBTTL TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
5433                                     ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
5434                                     ;*LEADING NUMBERS.
5435                                     ;*CALL
5436                                     ;*      MOV      #NUMADR, -(SP)      ;; FIRST ADDRESS OF ASCIZ STRING
5437                                     ;*      JSR      PC, @#SSUPRS
5438
5439
5440
5441
5442 021414 010046          $SUPRS: MOV     RO, -(SP)      ;; SAVE RO
5443 021416 016600 000004  MOV     4(SP),RO          ;; PICKUP THE POINTER
5444 021422 105710          1$:    TSTB   (RO)          ;; TERMINATOR?
5445 021424 001403          BEQ    2$              ;; BR IF YES
5446 021426 122720 000060  CMPB   #'0,(R0)+        ;; IS THIS AN ASCII "0" ?
5447 021432 001773          BEQ    1$              ;; BR IF YES
5448 021434 005300          2$:    DEC     RO          ;; BACKUP BY "1"
5449 021436 010037 021444  MOV     RO,3$           ;; SAVE FOR TYPING
5450 021442 104400          TYPE   GO TYPE        ;; GO TYPE
5451 021444 000000          3$:    .WORD 0          ;; ASCIZ POINTER GOES HERE
5452 021446 012600          MOV     (SP)+,RO        ;; RESTORE RO
5453 021450 012616          MOV     (SP)+,(SP)      ;; RESTORE THE STACK
5454 021452 000207          RTS     PC            ;; RETURN

```


5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510

.SBTTL INTEGER DIVIDE ROUTINE

;;THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
;;DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
;;A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
;;DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
;;SAVE SIGN AS THE DIVIDEND.

;;CALL:
;; MOV LOW DIVIDEND,-(SP) ;;THE HIGH DIVIDEND MUST BE < 1/2
;; MOV HIGH DIVIDEND,-(SP); AS LARGE AS THE DIVISOR
;; MOV DIVISOR,-(SP)
;; JSR PC,\$DIV
;; RETURN ;;QUOTIENT & REMAINDER ARE ON THE STACK
;; "V"=0 IMPLIES NO ERROR
;; "V"=1 IMPLIES ERROR OCCURRED
;; "C"=0 DIVIDE OVERFLOW OCCURRED
;; "C"=1 ATTEMPTED TO DIVIDE BY ZERO

;; STACK NO ERROR OVERFLOW DIVIDE BY ZERO
;; -----
;; TOP REMAINDER ALL ZEROS ALL ONES
;; +2 QUOTIENT ALL ZEROS ALL ONES

\$DIV: MOV 2#PS,-(SP) ;;SAVE THE PROCESSOR STATUS
BIC #17,(SP) ;;STRIP AWAY CONDITION CODES
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
CLR -(SP) ;;SAVE A PLACE FOR SIGNS
MOV #17,-(SP) ;;SETUP THE ITERATION COUNTER
MOV 24(SP),R1 ;;PICKUP THE DIVIDEND
MOV 22(SP),R0
BPL 1\$;;CHECK THE SIGN
DECB 3(SP) ;;KEEP TRACK OF THE SIGN
NEG R0 ;;AND NEGATE THE ORIGINAL
NEG R1 ;;NUMBER
SBC R0
1\$: MOV 20(SP),R2 ;;PICKUP THE DIVISOR
BLT 2\$;;CHECK THE SIGN
BGT 3\$;;DIVISOR OF 0 IS A NO-NO
BIS #3,14(SP) ;;SET "V" & "C"
MOV #-1,R0 ;;SET REMAINDER TO ALL ONES
BR 7\$;;EXIT
2\$: INC 2(SP) ;;KEEP TRACK OF DIVISORS SIGN
BR 4\$
3\$: NEG R2 ;;NEGATE THE ORIGINAL NUMBER
4\$: CLC ;;CLEAR "C"
BR 6\$;;START FORMING QUOTIENT
5\$: ROL R0 ;;POSITION MSB'S
MOV R0,R3 ;;COPY
ADD R2,R3 ;;COMPARE DIVIDEND & DIVISOR
BCC 6\$;;BR IF DIVIDEND > DIVISOR

021454 013746 177776
021460 042716 000017
021464 010046
021466 010146
021470 010246
021472 010346
021474 005046
021476 012746 000021
021502 016601 000024
021506 016600 000022
021512 100005
021514 105366 000003
021520 005400
021522 005401
021524 005600
021526 016602 000020
021532 002407
021534 003011
021536 052766 000003 000014
021544 012700 177777
021550 000424
021552 005266 000002
021556 000401
021560 005402
021562 000241
021564 000405
021566 006100
021570 010003
021572 060203
021574 !03001

```

5511 021576 010300          MOV      R3,RO          ;;REMAINDER AFTER THIS LOOP
5512 021600 006101          6$:    ROL      R1          ;;QUOTIENT BIT ENTERS HERE
5513 021602 005316          DEC      (SP)          ;;DONE?
5514 021604 001370          BNE     5$             ;;BR IF NO
5515 021606 005701          TST     R1             ;;OVERFLOW?
5516 021610 100005          BPL     8$             ;;BR IF NO
5517 021612 052766 000002 000014  BIS     #2,14(SP)      ;;SET "V" IN RETURN STATUS WORD
5518 021620 005000          CLR     RO             ;;SET REMAINDER TO ALL ZEROS
5519 021622 010001          7$:    MOV     RO,R1     ;;COPY REMAINDER INTO QUOTIENT
5520 021624 005726          8$:    TST     (SP)+     ;;CLEAR COUNTER FROM STACK
5521 021626 005716          TST     (SP)          ;;REMAINDER SIGN CORRECTION NEEDED?
5522 021630 002004          BGE     9$             ;;BR IF NO
5523 021632 005400          NEG     RO             ;;NEGATE REMAINDER
5524 021634 105066 000001  CLRB   1(SP)          ;;CLEAR SIGN
5525 021640 005316          DEC     (SP)          ;;BUT DON'T FORGET QUOTIENT
5526 021642 005726          9$:    TST     (SP)+     ;;QUOTIENT SIGN CORRECTION NEEDED?
5527 021644 001401          BEQ     10$           ;;BR IF NO
5528 021646 005401          NEG     R1             ;;NEGATE QUOTIENT
5529 021650 010166 000020 10$:   MOV     R1,20(SP)    ;;RETURN QUOTIENT AND
5530 021654 010066 000016  MOV     RO,16(SP)    ;;REMAINDER TO USER
5531 021660 012603          MOV     (SP)+,R3     ;;POP STACK INTO R3
5532 021662 012602          MOV     (SP)+,R2     ;;POP STACK INTO R2
5533 021664 012601          MOV     (SP)+,R1     ;;POP STACK INTO R1
5534 021666 012600          MOV     (SP)+,RO     ;;POP STACK INTO RO
5535 021670 012666 000002  MOV     (SP)+,2(SP)  ;;SETUP TO RETURN CONDITION CODES
5536 021674 000002          RTI                    ;;RETURN
5537 ;*****
5538 ;*****
5539 .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
5540
5541 ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
5542 ;*WITH A RANGE OF 0 TO 2(+33)-1.
5543 ;*CALL:
5544 ;*      JSR      PC,$RAND          ;;CALL THE ROUTINE
5545 ;*      RETURN                    ;;RETURN HERE THE RANDOM
5546 ;*                                ;;NUMBER WILL BE IN
5547 ;*                                ;;$HINUM,$LONUM
5548 ;*
5549 $RAND:
5550 021676 010046          MOV     RO,-(SP)     ;;PUSH RO ON STACK
5551 021700 010146          MOV     R1,-(SP)     ;;PUSH R1 ON STACK
5552 021702 010246          MOV     R2,-(SP)     ;;PUSH R2 ON STACK
5553 021704 010346          MOV     R3,-(SP)     ;;PUSH R3 ON STACK
5554 021706 013700 022024  MOV     $LONUM,R0    ;;SET RO WITH LOW
5555 021712 013701 022022  MOV     $HINUM,R1    ;;SET R1 WITH HIGH
5556 021716 012703 177771  MOV     #-7,R3      ;;SET SHIFT COUNT
5557 021722 005002          CLR     R2             ;;ZERO R2
5558 021724 006300          1$:    ASL     RO             ;;SHIFT RO LEFT AND
5559 021726 006101          ROL     R1             ;;ROTATE CARRY INTO R1 AND
5560 021730 006102          ROL     R2             ;;ROTATE CARRY INTO R2
5561 021732 005203          INC     R3             ;;CHECK FOR DONE
5562 021734 001373          BNE     1$             ;;CONTINUE SHIFT LOOP
5563 021736 063700 022024  ADD     $LONUM,RO     ;;ADD NUMBER TO MAKE X 129
5564 021742 005501          ADC     R1             ;;PROPOGATE CARRY
5565 021744 063701 022022  ADD     $HINUM,R1     ;;ADD NUMBER TO MAKE X 129
5566 021750 005502          ADC     R2             ;;PROPOGATE CARRY

```



```

5567 021752 062700 001057      ADD    #1057,R0      ;;ADD LOW CONSTANT
5568 021756 005501              ADC    R1            ;;PROPOGATE CARRY
5569 021760 005502              ADC    R2            ;;PROPOGATE CARRY
5570 021762 062701 047401      ADD    #47401,R1    ;;ADD HIGH CONSTANT
5571 021766 005502              ADC    R2            ;;PROPOGATE CARRY
5572 021770 062702 000006      ADD    #6,R2        ;;ADD HIGHEST CONSTART
5573 021774 060200              ADD    R2,R0        ;;REPRIME R0 WITH HIGHEST DIGIT
5574 021776 005501              ADC    R1            ;;PROPOGATE CARRY
5575 022000 010037 022024      MOV    RO,$LONUM    ;;SAVE RO
5576 022004 010137 022022      MOV    R1,$HINUM    ;;SAVE R1
5577 022010 012603              MOV    (SP)+,R3     ;;POP STACK INTO R3
5578 022012 012602              MOV    (SP)+,R2     ;;POP STACK INTO R2
5579 022014 012601              MOV    (SP)+,R1     ;;POP STACK INTO R1
5580 022016 012600              MOV    (SP)+,R0     ;;POP STACK INTO R0
5581 022020 000207              RTS    PC            ;;RETURN
5582 022022 176543      $HINUM: .WORD 176543
5583 022024 123456      $LONUM: .WORD 123456

```

.SBTTL ROUTINE TO SIZE MEMORY

```

5588 ;*CALL:
5589 ;*      JSR    PC,$SIZE
5590 ;*      RETURN
5591 ;*$LSTAD WILL CONTAIN:
5592 ;*      WITH KT11 OPTION      -- LAST VIRTUAL ADDRESS OF THE LAST BANK
5593 ;*      WITHOUT KT11 OPTION  -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
5594 ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
5595 ;*$KT11 IS THE MEMORY MANAGEMENT KEY
5596 ;*$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
5597 ;*      MUST BE SETUP BEFORE THE CALL
5598 ;*$BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
5599 ;*      DETERMINED BY ROUTINE

```

```

5601 022026 010046      $SIZE: MOV    RO,-(SP)    ;;SAVE RO ON THE STACK
5602 022030 010146      MOV    R1,-(SP)    ;;SAVE R1 ON THE STACK
5603 022032 010246      MOV    R2,-(SP)    ;;SAVE R2 ON THE STACK
5604 022034 010346      MOV    R3,-(SP)    ;;SAVE R3 ON THE STACK
5605 022036 013746 000004      MOV    @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
5606 022042 013746 000006      MOV    @#ERRVEC+2,-(SP)
5607 022046 010600      MOV    SP,RO       ;;SAVE THE STACK POINTER
5608 022050 013737 177776 000006      MOV    @#PS,@#ERRVEC+2 ;;SET ERRVEC PS TO PRESENT PS
5609 022056 012701 003776      MOV    #3776,R1    ;;SETUP ADDRESS
5610 022062 105727      TSTB   (PC)+       ;;USE MEMORY MANAGEMENT?
5611 022064 000200      $KT11: .WORD 200   ;;SET TO USE MEMORY MANAGEMENT
5612 022066 100063      BPL    $CORE       ;;BR IF NO
5613 022070 012737 022230 000004      MOV    #SKTNEX,@#ERRVEC ;;SET FOR TIMEOUT
5614 022076 005737 177572      TST    @#SRO       ;;KT11 ARE YOU THERE?
5615 022102 052737 100000 022064      BIS    #100000,$KT11 ;;YES--SET KT11 KEY
5616 022110 012737 022336 000250      MOV    #MMER,@#MMVEC ;;SET IN CASE OF ERROR
5617 022116 012737 000340 000252      MOV    #340,@#MMVEC+2
5618 022124 005046      CLR    -(SP)       ;;INITIALIZE FOR "PAR" LOADING
5619 022126 012702 172340      MOV    #KIPAR0,R2  ;;ADDRESS OF FIRST "PAR"
5620 022132 012703 000010      MOV    #↑DB,R3     ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
5621 022136 012762 077406 177740 1$: MOV    #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
5622 022144 011622      MOV    (SP),(R2)+  ;;LOAD "PAR"

```

```

5623 022146 062716 000200          ADD    #200,(SP)          ;;UPDATE FOR NEXT "PAR"
5624 022152 077307                   SOB    R3,1$            ;;LOOP UNTIL ALL EIGHT ARE LOADED
5625 022154 012742 177600          MOV    #177600,-(R2)    ;;SETUP KIPAR7 FOR I/O
5626 022160 005042                   CLR    -(R2)           ;;SETUP KIPAR6 FOR TESTING
5627 022162 012737 000020 172516  MOV    #20,@#SR3       ;;ENABLE 22 BIT MODE
5628 022170 005237 177572          INC    @#SR0           ;;TURN ON MEMORY MANAGEMENT
5629 022174 012737 022220 000004  MOV    #SKTOUT,@#ERRVEC ;;SET FOR TIME OUT
5630 022202 005737 143776          TST   @#143776        ;;TRAP ON NON-EX-MEM
5631 022206 062712 000040          ADD    #40,(R2)        ;;MAKE A 1K STEP
5632 022212 023712 172356          CMP    @#KIPAR7,(R2)   ;;LAST ONE?
5633 022216 101371                   BHI   2$              ;;NO--TRY IT
5634 022220 011202          SKTOUT: MOV    (R2),R2    ;;GET LAST BANK+1
5635 022222 005037 177572          CLR    @#SR0           ;;TURN OFF MEMORY MANAGEMENT
5636 022226 000421                   BR    $SIZEX
5637 022230 042737 100000 022064 SKTNEX: BIC    #100000,$KT11 ;;KT11 NON-EXISTENT
5638 022236 012737 022266 000004 SCORE: MOV    #SCROUT,@#ERRVEC ;;SET FOR TIMEOUT
5639 022244 005002                   CLR    R2              ;;SET UP BANK
5640 022246 062701 004000          1$:  ADD    #4000,R1    ;;INCREMENT BY 1K
5641 022252 062702 000040          ADD    #40,R2          ;;1K STEP
5642 022256 005711                   TST   (R1)            ;;TRAP ON TIME OUT
5643 022260 022701 177776          CMP    #177776,R1     ;;LAST ONE
5644 022264 001370                   BNE   1$              ;;NO--TRY AGAIN
5645 022266 162701 004000          SCROUT: SUB    #4000,R1
5646 022272 162702 000040          $SIZEX: SUB    #40,R2  ;;DROP BACK
5647 022276 010006                   MOV    R0,$P          ;;RESTORE THE STACK
5648 022300 012637 000006          MOV    ($P)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
5649 022304 012637 000004          MOV    ($P)+,@#ERRVEC
5650 022310 010137 022332          MOV    R1,$LSTAD      ;;LAST ADDRESS
5651 022314 010237 022334          MOV    R2,$LSTBK      ;;LAST BANK
5652 022320 012603                   MOV    ($P)+,R3        ;;RESTORE R3
5653 022322 012602                   MOV    ($P)+,R2        ;;RESTORE R2
5654 022324 012601                   MOV    ($P)+,R1        ;;RESTORE R1
5655 022326 012600                   MOV    ($P)+,R0        ;;RESTORE R0
5656 022330 000207          RTS    PC
5657 022332 000000          $LSTAD: .WORD    0      ;;CONTAINS THE LAST ADDRESS
5658 022334 000000          $LSTBK: .WORD    0      ;;CONTAINS THE LAST BANK
5659 022336 000000          MMER:  HALT
5660
5661          ;*****
5662          .SBTTL LP.AVL - LINE PRINTER AVAILABLE
5663          ;*THIS ROUTINE WILL CHECK IF THERE IS A LINE PRINTER AVAILABLE AND SET
5664          ;*"LPTAVL" TO THE PROPER STATE.
5665          ;* LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
5666          ;* LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
5667          ;*CALL
5668          ;*   JSR    PC,@#LP.AVL
5669          ;*   RETURN
5670
5671 022340 005037 001254          LP.AVL: CLR    @#LPTAVL    ;;START WITH NO PRINTER AVAIBLE
5672 022344 012737 022370 000004  MOV    #1$,@#ERRVEC    ;;SETUP THE TIMEOUT VECTOR
5673 022352 005037 000006          CLR    @#ERRVEC+2
5674 022356 005777 157032          TST   @LPS            ;;IS THERE A LINE PRINTER?
5675 022362 005237 001254          INC    @#LPTAVL       ;;YES--SET AVAILABLE SWITCH
5676 022366 000401                   BR    2$
5677 022370 022626          1$:  CMP    ($P)+,($P)+    ;;NO--POP STACK
5678 022372 012737 000006 000004  2$:  MOV    #ERRVEC+2,@#ERRVEC ;;RESTORE TIMEOUT VECTOR

```



```

5679 022400 000207          RTS      PC          ;RETURN
5680
5681 ;*****
5682 .SBTTL ST.CLK - CLOCK STARTUP ROUTINE
5683 ;*THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
5684 ;*AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
5685 ;*"CLKSTA" WILL INDICATE THE CLOCK TYPE
5686 ;* 0= NO CLOCK
5687 ;*+1= KW11-P
5688 ;*-1= KW11-L
5689 ;*THIS ROUTINE WILL ALSO SETUP "TICKMS" (TIME
5690 ;*PER CLOCK TICK IN MILLISECONDS) AND "TICKUS"
5691 ;*(TIME PER CLOCK TICK IN MICROSECONDS) AS
5692 ;*PER SW00.
5693 ;*SW00=0 -- 60HZ
5694 ;*SW00=1 -- 50HZ
5695 ;*CALL
5696 ;*      JSR      PC,@#ST.CLK
5697 ;*      RETURN
5698
5699 022402 010146          ST.CLK: MOV      R1,-(SP)          ;SAVE R1
5700 022404 012701 000006  MOV      #ERRVEC+2,R1      ;SAVE AND SETUP TIMEOUT VECTOR
5701 022410 011146          MOV      (R1),-(SP)
5702 022412 005011          CLR      (R1)              ;LEVEL 0
5703 022414 014146          MOV      -(R1),-(SP)
5704 022416 012711 022446  MOV      #1$, (R1)          ;GO TO 1$ ON TIMEOUT
5705 022422 005037 001262  CLR      CLKSTA           ;SET CLOCK STATUS TO NO CLOCK
5706 022426 005777 156742  TST      @PKCS             ;IS THERE A KW11-P?
5707 022432 012737 000001 001262  MOV      #1,CLKSTA        ;YES--SET STATUS TO KW11-P
5708 022440 004737 022550  JSR      PC,ST.PCLK       ;START THE KW11-P
5709 022444 000414          BR      3$                ;GO TO EXIT
5710 022446 022626          1$:  CMP      (SP)+,(SP)+      ;CLEAN UP THE STACK
5711 022450 012711 022474  MOV      #2$, (R1)          ;IF TIMEOUT GO TO 2$
5712 022454 005777 156726  TST      @LK$             ;IS THERE A KW11-L?
5713 022460 012737 177777 001262  MOV      #-1,CLKSTA       ;YES-- SET STATUS TO KW11-L
5714 022466 004737 022612  JSR      PC,ST.LCLK       ;START THE KW11-L
5715 022472 000401          BR      3$                ;EXIT
5716 022474 022626          2$:  CMP      (SP)+,(SP)+      ;CLEAN UP THE STACK
5717 022476 012621 3$:  MOV      (SP)+,(R1)+      ;RESTORE THE TIMEOUT VECTOR
5718 022500 012621          MOV      (SP)+,(R1)+
5719 022502 012601          MOV      (SP)+,R1         ;RESTORE R1
5720 022504 032737 000100 001246  BIT      #SW06,@#C.SWR     ;50HZ OR 60HZ?
5721 022512 001407          BEQ     4$                ;BRANCH IF 60
5722 022514 012737 000020 001264  MOV      #20,@#TICKMS      ;SETUP TIME PER
5723 022522 012737 047040 001266  MOV      #20000.,@#TICKUS ;TICK FOR 50HZ
5724 022530 000406          BR      5$
5725 022532 012737 000016 001264 4$:  MOV      #16,@#TICKMS     ;SETUP TIME PER
5726 022540 012737 040432 001266  MOV      #16666.,@#TICKUS ;TICK FOR 60HZ
5727 022546 000207          5$:  RTS      PC              ;RETURN
5728
5729 022550          ST.PCLK:
5730 022550 032737 000040 001246  BIT      #SW05,@#C.SWR     ;ALLOW SOFTWARE TIMEOUTS?
5731 022556 001014          BNE     1$                ;NO--BRANCH
5732 022560 012777 022646 156602  MOV      #SRVCLK,@PKV     ;SETUP THE KW11-P VECTOR
5733 022566 012777 000300 156576  MOV      #300,@PKV+2
5734 022574 012777 000001 156574  MOV      #1,@PKB          ;COUNT ONE TICK

```


B10

```

5735 022602 012777 000115 156564      MOV      #115,DPKCS      ;"INT.EN." COUNT DOWN" "MODE 1 (REPEAT)",
5736                                     ;"LINE FREQ", AND "RUN"
5737 022610 000207      1$:      RTS      PC      ;RETURN
5738
5739 022612      ST.LCLK:
5740 022612 032737 000040 001246      BIT      #SW05,DP.C.SWR ;ALLOW SOFTWARE TIMEOUTS?
5741 022620 001011      BNE      1$          ;NO--BRANCH
5742 022622 012777 022646 156552      MOV      #SRVCLK,DLKV  ;SETUP THE KW11-L VECTOR
5743 022630 012777 000300 156546      MOV      #300,DLKV+2
5744 022636 012777 000100 156542      MOV      #100,DLKS    ;START THE KW11-L
5745 022644 000207      1$:      RTS      PC      ;RETURN
5746
5747 022646 013746 001264      SRVCLK: MOV      DP.TICKMS,-(SP) ;TIME PER TICK.IN MILLISECONDS
5748 022652 004037 036144      JSR      RO,DP.RPTMR  ;COUNT THE ELAPSED TIME
5749 022656 000002      RTI          ;RETURN AFTER INTERRUPT
5750
5751      ;*****
5752      .SBTTL LDCMD - LOAD COMMAND
5753      ;*THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
5754      ;*INTO DPB.B+2 AND DPB.C+2, DEPENDING ON THE STATE OF "CONTROL SWITCH"
5755      ;*BIT07.
5756      ;*CALL
5757      ;*      JSR      PC,DP.LDCMD
5758      ;*      RETURN
5759
5760 022660 032737 000200 001246 LDCMD: BIT      #SW07,DP.C.SWR ;DO EXPLICIT SEEKS?
5761 022666 001007      BNE      1$          ;YES--BRANCH
5762 022670 012737 000173 003516      MOV      #READHD,DPB.B+2 ;NO--SET UP FOR READ HEADER AND
5763 022676 012737 000173 003536      MOV      #READHD,DPB.C+2 ;DATA COMMAND
5764 022704 000406      BR      2$
5765 022706 012737 000105 003516 1$:      MOV      #SEEK,DPB.B+2 ;SETUP FOR SEEK COMMAND
5766 022714 012737 000105 003536      MOV      #SEEK,DPB.C+2
5767 022722 000207      2$:      RTS      PC
5768
5769      ;*****
5770      .SBTTL CALL.A - CALL RPO4 DRIVER USING "DPB.A"
5771      ;*THIS ROUTINE WILL CALL THE RPO4 DRIVER AND THEN WAIT ON THE FUNCTION
5772      ;*TO COMPLETE. IF AN ERROR OCCURS IT IS REPORTED.
5773      ;*CALL
5774      ;*      FILL "DPB" WITH COMMAND INFORMATION
5775      ;*      JSR      RO,DP.CALL.A
5776      ;*      RETURN
5777
5778 022724 005037 001200      CALL.A: CLR      DP.$ESCAPE ;NO ESCAPE ADDRESS
5779 022730 004037 032766      JSR      RO,DP.RPO4  ;CALL RPO4 DRIVER
5780 022734 003474      DPB.A
5781 022736 000772      BR      CALL.A
5782 022740 005737 003512 1$:      TST      DPB.A+16 ;DONE?
5783 022744 001775      BEQ      1$        ;NO--LOOP
5784 022746 100030      BPL      3$        ;BRANCH IF NO ERROR
5785 022750 012737 023024 001200      MOV      #2$,DP.$ESCAPE ;ESCAPE TO 2$ ON ERROR
5786 022756 013737 003506 001306      MOV      DPB.A+12,DP.CYL.DS ;CYLINDER
5787 022764 113737 003505 001312      MOVB    DPB.A+11,DP.TRK.DS ;TRACK
5788 022772 113737 003504 001310      MOVB    DPB.A+10,DP.SEC.DS ;SECTOR
5789 023000 012746 003512      MOV      DPB.A+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
5790 023004 004737 024026      JSR      PC,DP.ERINDX ;FORM DISPATCH INDEX

```



```

5791 023010 062607          ADD      (SP)+,PC          ;REPORT PROPER ERROR
5792 023012 104041          ERROR    41              ;
5793 023014 104042          ERROR    42              ;PARITY ERROR
5794 023016 104043          ERROR    43              ;UNSAFE ERROR
5795 023020 104044          ERROR    44              ;NON-I/O ERROR
5796 023022 104045          ERROR    45              ;I/O ERROR
5797 023024 013700 001270 2$: MOV      @#BYPASS,RO      ;TAKE ERROR EXIT
5798 023030 000200          RTS        RO            ;RETURN
5799
5800
5801 ;*****
5802 ;SBTTL CALL.B - CALL RPO4 DRIVER USING "DPB.B"
5803 ;*THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND IF
5804 ;*THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,
5805 ;*AND SECTOR) READ IS CHECKED FOR VALIDITY.
5806 ;*CALL
5807 ;*   FILL DPB
5808 ;*   JSR      RO,@#CALL.B
5809 ;*   RETURN
5810 023032 005037 001200  CALL.B: CLR      @#$ESCAPE      ;NO ESCAPE ADDRESS
5811 023036 004037 032766  JSR      RO,@#RPO4      ;CALL RPO4 DRIVER
5812 023042 003514          DPB.B
5813 023044 000772          BR        CALL.B
5814 023046 005737 003532  1$: TST      DPB.B+16      ;DONE?
5815 023052 001775          BEQ      1$             ;NO--BRANCH
5816 023054 100031          BPL      3$             ;BRANCH IF NO ERROR
5817 023056 012737 023132 001200  MOV      @2$,$ESCAPE      ;ESCAPE TO 2$ ON ERROR
5818 023064 013737 003526 001306  MOV      @#DPB.B+12,@#CYL.DS ;CYLINDER
5819 023072 113737 003525 001312  MOV      @#DPB.B+11,@#TRK.DS ;TRACK
5820 023100 113737 003524 001310  MOV      @#DPB.B+10,@#SEC.DS ;SECTOR
5821 023106 012746 003532          MOV      @#DPB.B+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
5822 023112 004737 024026          JSR      PC,@#ERINDX     ;FORM DISPATCH INDEX
5823 023116 062607          ADD      (SP)+,PC          ;REPORT PROPER ERROR
5824 023120 104041          ERROR    41              ;
5825 023122 104042          ERROR    42              ;PARITY ERROR
5826 023124 104043          ERROR    43              ;UNSAFE ERROR
5827 023126 104044          ERROR    44              ;NON-I/O ERROR
5828 023130 104045          ERROR    45              ;I/O ERROR
5829 023132 013700 001270 2$: MOV      @#BYPASS,RO      ;TAKE ERROR EXIT
5830 023136 000407          BR        4$
5831 023140 123727 003516 000173 3$: CMPB    @#DPB.B+2,@#READHD ;DOING IMPLIED SEEKS?
5832 023146 001003          BNE      4$             ;NO--BRANCH
5833 023150 004037 024306          JSR      RO,@#VERIFY     ;YES--GO CHECK THE DATA
5834 023154 003524          DPB.B+10
5835 023156 032737 040000 001246 4$: BIT      @SW14,@#C.SWR   ;STALL?
5836 023164 001403          BEQ      5$             ;NO--BRANCH
5837 023166 004037 024144          JSR      RO,@#STALL     ;YES--CALL STALL ROUTINE
5838 023172 001420          .WORD   STALL1         ;STALL TIME POINTER
5839 023174 000200          RTS        RO            ;RETURN
5840
5841 ;*****
5842 ;SBTTL CALL.C - CALL RPO4 DRIVER USING "DBP.C"
5843 ;*THIS ROUTINE IS THE SAME AS "CALL.B" EXCEPT FOR THE DPB USED.
5844 ;*CALL
5845 ;*   FILL DPB
5846 ;*   JSR      RO,@#CALL.C

```

```

5847 ;* RETURN
5848
5849 023176 005037 001200 CALL.C: CLR @#SESCAPE ;NO ESCAPE ADDRESS
5850 023202 004037 032766 JSR RO,@#RPO4 ;CALL RPO4 DRIVER
5851 023206 003534 DPB.C
5852 023210 000772 BR CALL.C
5853 023212 005737 003552 1$: TST @#DPB.C+16 ;DONE?
5854 023216 001775 BEQ 1$ ;NO--LOOP
5855 023220 100031 BPL 3$ ;YES--BRANCH IF NO ERROR
5856 023222 012737 023276 001200 MOV #2$,SESCAPE ;ESCAPE TO 2$ ON ERROR
5857 023230 013737 003546 001306 MOV @#DPB.C+12,@#CYL.DS ;CYLINDER
5858 023236 113737 003545 001312 MOVB @#DPB.C+11,@#TRK.DS ;TRACK
5859 023244 113737 003544 001310 MOVB @#DPB.C+10,@#SEC.DS ;SECTOR
5860 023252 012746 003552 MOV @#DPB.C+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
5861 023256 004737 024026 JSR PC,@#ERINDX ;FORM DISPATCH INDEX
5862 023262 062607 ADD (SP)+,PC ;REPORT PROPER ERROR
5863 023264 104041 ERROR 41 ;
5864 023266 104042 ERROR 42 ;PARITY ERROR
5865 023270 104043 ERROR 43 ;UNSAFE ERROR
5866 023272 104044 ERROR 44 ;NON-I/O ERROR
5867 023274 104045 ERROR 45 ;I/O ERROR
5868 023276 013700 001270 2$: MOV @#BYPASS,RO ;TAKE ERROR EXIT
5869 023302 000400 BR 3$
5870 023304 123727 003536 000173 3$: CMPB @#DPB.C+2,@#READHD ;DOING IMPLIED SEEK?
5871 023312 001003 BNE 4$ ;NO--EXIT
5872 023314 004037 004306 JSR RO,@#VERIFY ;YES--CHECK THE DATA
5873 023320 003544 DPB.C+10
5874 023322 032737 040000 001246 4$: BIT #SW14,@#C.SWR ;STALL?
5875 023330 001403 BEQ 5$ ;NO--BRANCH
5876 023332 004037 024144 JSR RO,@#STALL ;YES--CALL STALL ROUTINE
5877 023336 001420 .WORD STALL1 ;STALL TIME POINTER
5878 023340 000200 5$: RTS RO

```

```

*****
.SBTTL DRVCAL - DRIVER (RPO4) CALL USING "DTADPB"
;THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND
;ON AN ERROR LOCATION "ERR.CT" IS EXAMINED. IF ERR.CT IS EQUAL TO
;#SERFLG EXIT IS TO THE NEXT TEST.
;CALL

```

```

;* FILL DPB
;* JSR RO,@#DRVCAL
;* RETURN

```

```

5891 023342 005037 001200 DRVCAL: CLR @#SESCAPE ;NO ESCAPE ADDRESS
5892 023346 005037 001352 CLR @#WCEFLG ;CLEAR WRITE CHECK ERROR FLAG
5893 023352 004037 032766 JSR RO,@#RPO4 ;CALL RPO4 DRIVER
5894 023356 003554 DTADPB
5895 023360 000770 BR DRVCAL
5896 023362 005737 003572 1$: TST @#DTADPB+16 ;DONE
5897 023366 001775 BEQ 1$ ;NO--LOOP
5898 023370 100001 BPL 12$ ;YES--BRANCH IF NO ERRORS
5899 023372 000402 BR 13$
5900 023374 000137 024006 12$: JMP 10$
5901 023400 13$:
5902 023400 012737 023454 001200 MOV #2$,SESCAPE ;;ESCAPE TO 2$ ON ERROR

```


E10

5903	023406	013737	003566	001306		MOV	@#DTADPB+12,@#CYL.DS	;CYLINDER
5904	023414	113737	003565	001312		MOVB	@#DTADPB+11,@#TRK.DS	;TRACK
5905	023422	113737	003564	001310		MOVB	@#DTADPB+10,@#SEC.DS	;SECTOR
5906	023430	012746	003572			MOV	@#DTADPB+16,-(SP)	;STATUS/ERROR INDICATOR ADDRESS
5907	023434	004737	024026			JSR	PC,@#ERINDX	;FORM DISPATCH INDEX
5908	023440	062607				ADD	(SP)+,PC	;REPORT PROPER ERROR
5909	023442	104041				ERROR	41	
5910	023444	104042				ERROR	42	;PARITY ERROR
5911	023446	104043				ERROR	43	;UNSAFE ERROR
5912	023450	104044				ERROR	44	;NON-I/O ERROR
5913	023452	104045				ERROR	45	;I/O ERROR
5914	023454	122737	000015	001102	2\$:	CMPB	#15,@#STSTNM	;TEST 15?
5915	023462	001137				BNE	8\$;NO--BRANCH
5916	023464	122737	000017	001102		CMPB	#17,@#STSTNM	;TEST 17 ? MINE
5917	023472	001133				BNE	8\$;NO--BRANCH
5918	023474	122737	000151	003556		CMPB	#WRCKD,@#DTADPB+2	;DOING A WRITE CHECK?
5919	023502	001127				BNE	8\$;NO--BRANCH
5920	023504	032737	040000	003604		BIT	#BIT14,@#RP.REG+10	;IS "WCE"=1?
5921	023512	001523				BEQ	8\$;NO--BRANCH
5922	023514	032737	000020	177570		BIT	#SW04,@#SWR	;INHIBIT WRITES?
5923	023522	001117				BNE	8\$;YES--BRANCH
5924	023524	112737	000161	003556		MOVB	#WRITE,@#DTADPB+2	;SETUP FOR A WRITE
5925	023532	005037	001200			CLR	@#\$ESCAPE	;NO ESCAPE ADDRESS
5926	023536	004037	032766			JSR	RO,@#RPO4	;DO THE WRITE
5927	023542	003554				DTADPB		
5928	023544	000240				NOP		
5929	023546	005737	003572		3\$:	TST	@#DTADPB+16	;DONE?
5930	023552	001775				BEQ	3\$;NO--LOOP
5931	023554	100026				BPL	4\$;YES--BRANCH IF NO ERROR
5932	023556	012737	023762	001200		MOV	#8\$, \$ESCAPE	;ESCAPE TO 8\$ ON ERROR
5933	023564	013737	003566	001306		MOV	@#DTADPB+12,@#CYL.DS	;CYLINDER
5934	023572	113737	003565	001312		MOVB	@#DTADPB+11,@#TRK.DS	;TRACK
5935	023600	113737	003564	001310		MOVB	@#DTADPB+10,@#SEC.DS	;SECTOR
5936	023606	012746	003572			MOV	@#DTADPB+16,-(SP)	;STATUS/ERROR INDICATOR ADDRESS
5937	023612	004737	024026			JSR	PC,@#ERINDX	;FORM DISPATCH INDEX
5938	023616	062607				ADD	(SP)+,PC	;REPORT PROPER ERROR
5939	023620	104041				ERROR	41	
5940	023622	104042				ERROR	42	;PARITY ERROR
5941	023624	104043				ERROR	43	;UNSAFE ERROR
5942	023626	104044				ERROR	44	;NON-I/O ERROR
5943	023630	104045				ERROR	45	;I/O ERROR
5944	023632	112737	000151	003556	4\$:	MOVB	#WRCKD,@#DTADPB+2	;COMMAND=WRITE CHECK DATA
5945	023640	004037	032766			JSR	RO,@#RPO4	;DO THE WRITE CHECK
5946	023644	003554				DTADPB		
5947	023646	000240				NOP		
5948	023650	005737	003572		5\$:	TST	@#DTADPB+16	;DONE?
5949	023654	001775				BEQ	5\$;NO--LOOP
5950	023656	100410				BMI	7\$;YES--BRANCH IF ERROR
5951	023660	004037	032766			JSR	RO,@#RPO4	;DO A 2ND WRITE CHECK
5952	023664	003554				DTADPB		
5953	023666	000240				NOP		
5954	023670	005737	003572		6\$:	TST	@#DTADPB+16	;DONE?
5955	023674	001775				BEQ	6\$;NO--LOOP
5956	023676	100031				BPL	8\$;YES--BRANCH IF NO ERROR
5957	023700	012737	000001	001352	7\$:	MOV	#1,@#WCEFLG	;SET THE WRITE CHECK ERROR FLAG
5958	023706	012737	023762	001200		MOV	#8\$, \$ESCAPE	;ESCAPE TO 8\$ ON ERROR

F10

```

5959 023714 013737 003566 001306      MOV      @#DTADPB+12,@#CYL.DS ;CYLINDER
5960 023722 113737 003565 001312      MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
5961 023730 113737 003564 001310      MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
5962 023736 012746 003572          MOV      #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
5963 023742 004737 024026      JSR      PC,@#ERINDX ;FORM DISPATCH INDEX
5964 023746 062607          ADD      (SP)+,PC ;REPORT PROPER ERROR
5965 023750 104041          ERROR   41 ;
5966 023752 104042          ERROR   42 ;PARITY ERROR
5967 023754 104043          ERROR   43 ;UNSAFE ERROR
5968 023756 104044          ERROR   44 ;NON-I/O ERROR
5969 023760 104046          ERROR   46 ;FATAL WRITE CHECK
5970 023762 032737 001000 177570 8$:      BIT      #SW09,@#SWR ;LOOP ON ERROR?
5971 023770 001004          BNE     9$ ;YES--BRANCH
5972 023772 123737 001426 001103      CMPB    @#ERR.CT,@#SERFLG ;GO TO NEXT TEST?
5973 024000 101002          BHI     10$ ;NO--BRANCH
5974 024002 013700 001270          MOV      @#BYPASS,RO ;YES--GET EXIT ADDRESS
5975 024006 032737 040000 001246 10$:      BIT      #SW14,@#C.SWR ;STALL?
5976 024014 001403          BEQ     11$ ;NO--BRANCH
5977 024016 004037 024144      JSR      RO,@#STALL ;YES--CALL STALL ROUTINE
5978 024022 001422          .WORD  STALL2 ;STALL TIME POINTER
5979 024024 000200          11$:    RTS      RO
  
```

.SBTTL ERINDX - FORM ERROR INDEX
 *THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
 *TO THE PROPER ERROR CALL. THE INDEX IS FORMED BY EXAMINING
 *THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.

```

*INDEX      STATUS/ERROR
*-----
*          0      BIT14!BIT13!BIT08
*          2      BIT11!BIT10
*          4      BIT12!BIT04
*          6      BIT05!BIT03!<BIT09 & COMMAND=NON-I/O>
*          10     BIT06!<BIT09 & COMMAND=I/O>
  
```

```

*CALL
*          JSR      #DPB+16,-(SP) ;ADDRESS OF STATUS/ERROR INDICATOR
*          JSR      PC,@#ERINDX ;FORM INDEX
*          RETURN ;INDEX IS ON THE STACK
  
```

```

5998 024026 010046          ERINDX: MOV      RO,-(SP) ;SAVE RO
5999 024030 010146          MOV      R1,-(SP) ;SAVE R1
6000 024032 016600 000006      MOV      6(SP),RO ;GET STATUS/ERROR INDICATOR POINTER
6001 024036 011037 001276      MOV      (RO),@#SVSTAT ;SAVE THE STATUS/ERROR INDICATOR
6002 024042 005001          CLR      R1 ;START INDEX AT ZERO
6003 024044 032710          BIT      (PC)+,(RO) ;FORM INDEX OF 0?
6004 024046 060400          .WORD  BIT14!BIT13!BIT08
6005 024050 001027          BNE     5$ ;YES--BRANCH
6006 024052 032710          BIT      (PC)+,(RO) ;FORM PARITY ERROR INDEX (2)?
6007 024054 006000          .WORD  BIT11!BIT10
6008 024056 001023          BNE     4$ ;YES--BRANCH
6009 024060 032710          BIT      (PC)+,(RO) ;FORM UNSAFE INDEX (4)?
6010 024062 010020          .WORD  BIT12!BIT04
6011 024064 001017          BNE     3$ ;YES--BRANCH
6012 024066 032710          BIT      (PC)+,(RO) ;FORM NON-I/O ERROR INDEX (6)?
6013 024070 000050          .WORD  BIT05!BIT03
6014 024072 001013          BNE     2$ ;YES--BRANCH
  
```


6015	024074	032710			BIT	(PC)+,(RO)	;FORM I/O ERROR INDEX (10)?
6016	024076	000100			.WORD	BIT06	
6017	024100	001007			BNE	1\$;YES--BRANCH
6018	024102	032710			BIT	(PC)+,(RO)	;SOFTWARE TIMEOUT?
6019	024104	001000			.WORD	BIT09	
6020	024106	001410			BEQ	5\$;NO--FORM INDEX OF 0
6021	024110	122760	000150	177762	CMPB	#150,-16(RO)	;YES--I/O?
6022	024116	003001			BGT	2\$;NO--BRANCH
6023	024120	005201		1\$:	INC	R1	;INDEX=10---ERROR=45 OR 46
6024	024122	005201		2\$:	INC	R1	;INDEX=6---ERROR=44
6025	024124	005201		3\$:	INC	R1	;INDEX=4---ERROR=43
6026	024126	005201		4\$:	INC	R1	;INDEX=2---ERROR=42
6027	024130	006301		5\$:	ASL	R1	;INDEX=0---ERROR=41
6028	024132	010166	000006		MOV	R1,6(SP)	;RETURN INDEX TO USER
6029	024136	012601			MOV	(SP)+,R1	;RESTORE R1
6030	024140	012600			MOV	(SP)+,RO	;RESTORE RO
6031	024142	000207			RTS	PC	;RETURN FROM CALL

```

;*****
;SBTTL STALL - USED TO STALL AFTER A DRIVE FUNCTION IS COMPLETE
;THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
;AMOUNT OF TIME IF BIT13 OF C.SWR = 0 OR A RANDOM AMOUNT OF TIME
;IF BIT 13 OF C.SWR = 1.
;STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0-6 AND STALL2
;CONTAINS THE TIME FOR TESTS 13-16.
;CALL

```

6041					* JSR	RO,2#STALL		
6042					* TIME POINTER		;WHERE TO FIND THE STALL TIME	
6043								
6044	024144	013046			STALL: MOV	2(RO)+,-(SP)	;PICKUP STALL TIME	
6045	024146	032737	020000	001246	BIT	#SW13,2#C.SWR	;USE A RANDOM TIME?	
6046	024154	001406			BEQ	1\$;NO--BRANCH	
6047	024156	004737	021676		JSR	PC,2#SRAND	;YES--FORM RANDOM NUMBER	
6048	024162	013716	022024		MOV	2#\$LONUM,(SP)	;AND USE IT FOR THE STALL TIME	
6049	024166	042716	177700		BIC	#1C77,(SP)	;BUT NEVER > 64 MILLISECONDS	
6050	024172	005046		1\$:	CLR	-(SP)	;CLEAR TEMP. LOCATION	
6051	024174	162766	000001	000002	2\$:	SUB	#1,2(SP)	;MORE STALL REQUIRED?
6052	024202	103407			BLO	4\$;NO--BRANCH	
6053	024204	012716	000144		MOV	#1D100,(SP)	;STALL FOR ABOUT 1 MILLISECOND	
6054	024210	005700		3\$:	TST	RO	;NOP TO KILL TIME	
6055	024212	005366	000000		DEC	0(SP)	;COUNT	
6056	024216	001374			BNE	3\$;LOOP IF MORE COUNTS NEEDED	
6057	024220	000765			BR	2\$		
6058	024222	022626		4\$:	CMP	(SP)+,(SP)+	;CLEAN OFF THE STACK	
6059	024224	000200			RTS	RO	;EXIT	

```

;*****
;SBTTL TWOMS - STALL FOR 2 MS BETWEEN SEEKS IN TESTS 10- - 12
;ROUTINE TO PROVIDE A 2 MS STALL AFTER A SEEK OPERATION IN THE SEEK TIMING
;TESTS (TESTS 10, 11, & 12). THIS STALL IS SPECIFIED BY THE VENDOR (ISS)
;OF THE DISK DRIVE. THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED
;SEEK TIMES.
;CALL
; JSR PC,2#TWOMS
; RETURN

```

6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070

```

6071
6072 024226 013746 177776 TWOMS: MOV @#PS,-(SP) ;SAVE THE PRESENT PROCESSOR STATUS
6073 024232 012737 000240 177776 MOV #<5*32.>,@#PS ;SET THE PROCESSOR PRIORITY TO 5
6074 024240 017746 155124 MOV @PKV,-(SP) ;SAVE THE OLD CLOCK VECTOR ADDRESS
6075 024244 012777 024270 155116 MOV #1$,@PKV ;SETUP NEW VECTOR ADDRESS
6076 024252 012777 000310 155116 MOV #200,@PKB ;LOAD THE CLOCK BUFFER
6077 024260 012777 000101 155106 MOV #101,@PKCS ;START THE CLOCK
6078 024266 000001 WAIT ;WAIT FOR 2 MS
6079 024270 062706 000004 1$: ADD #4,SP ;INCREMENT STACK FOR RETURN
6080 024274 012677 155070 MOV (SP)+,@PKV ;RESTORE OLD CLOCK VECTOR
6081 024300 012637 177776 MOV (SP)+,@#PS ;RESTORE THE OLD PROCESSOR STATUS
6082 024304 000207 RTS PC ;RETURN
6083
6084 ;*****
6085 ;SBTTL VERIFY - VALIDATE HEADER ON IMPLIED SEEKS
6086 ;*ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS
6087 ;*CALL
6088 ;* JSR RO,@#VERIFY ;ADDRESS OF DPB+10 (SECTOR NUMBER)
6089 ;* ADR POINTER
6090 ;* RETURN
6091
6092 024306 010146 VERIFY: MOV R1,-(SP) ;SAVE R1
6093 024310 012001 MOV (R0)+,R1 ;GET ADDRESS OF DPB+10
6094 024312 042737 010000 044134 BIC #FMT22,@#BUFFER ;STRIP FORMAT BIT FROM CYLINDER NUMBER
6095 024320 023761 044134 000002 CMP @#BUFFER,2(R1) ;CYLINDER NUMBER OK?
6096 024326 001003 BNE 1$ ;NO--BRANCH
6097 024330 023711 044136 CMP @#BUFFER+2,(R1) ;YES--HOW ABOUT TRACK/SECTOR?
6098 024334 001426 BEQ 3$ ;BRANCH IF GOOD
6099 024336 013737 044134 001300 1$: MOV @#BUFFER,@#CYL.RD ;SAVE THE EXPECTED AND THE
6100 024344 113737 044137 001302 MOVB @#BUFFER+3,@#TRK.RD ;RECIEVED CYLINDER, TRACK,
6101 024352 113737 044136 001304 MOVB @#BUFFER+2,@#SEC.RD ;AND SECTOR
6102 024360 112137 001310 MOVB (R1)+,@#SEC.DS
6103 024364 112137 001312 MOVB (R1)+,@#TRK.DS
6104 024370 011137 001306 MOV (R1),@#CYL.DS
6105 024374 012737 024406 001200 MOV #2$, $ESCAPE ;:ESCAPE TO 2$ ON ERROR
6106 024402 005740 TST -(R0) ;MAKE IT TEST PC+4
6107 024404 104012 ERROR 12 ;REPORT THE ERROR
6108 024406 013700 001270 2$: MOV @#BYPASS,R0 ;TAKE ERROR EXIT
6109 024412 012601 3$: MOV (SP)+,R1 ;RESTORE R1
6110 024414 000200 RTS R0 ;EXIT
6111
6112 ;*****
6113 ;SBTTL SRCH00 - INITIALIZE THE DRIVE FOR THE TIMING TESTS
6114 ;*THIS ROUTINE WILL PERFORM A "MASSBUS" INIT. FOLLOWED BY
6115 ;*A "RECALIBRATE" ON THE DRIVE UNDER TEST.
6116 ;*NOTE: THIS ROUTINE DESTROYS R1 AND R4
6117 ;*CALL
6118 ;* JSR RO,SRCH00 ;DO A MASSBUS INIT. AND RECAL
6119 ;* RETURN1 ;RETURN HERE IF NO ERROR
6120 ;* RETURN2 ;RETURN HERE ON ERROR
6121
6122 024416 005001 SRCH00: CLR R1 ;INCASE OF ERROR (TYPTIM)
6123 024420 005037 177776 CLR @#PS
6124 024424 012777 034766 005730 MOV #ISR,@RPVEC ;SETUP INTERRUPT VECTOR
6125 024432 013704 032360 MOV @#RPADR,R4 ;PICKUP ADDRESS OF RHCS1
6126 024436 012764 000040 000010 MOV #BIT05,RHCS2(R4) ;MASSBUS INIT.

```



```

6127 024444 005037 003564 CLR      @#DTADPB+10      ;TRACK=0. SECTOR=0
6128 024450 005037 003566 CLR      @#DTADPB+12      ;CYLINDER =0
6129 024454 012737 000107 003556 MOV      #RECAL,@#DTADPB+2 ;COMMAND = RECALIBRATE
6130 024462 005037 001200 CLR      @#$ESCAPE        ;NO ESCAPE ADDRESS
6131 024466 004037 032766 JSR      RD,@#RPO4        ;CALL THE DRIVER
6132 024472 003554 DTADPB    ;DPB POINTER
6133 024474 000440 BR        4$              ;QUEUE IS FULL
6134 024476 005737 003572 1$: TST     DTADPB+16      ;WAIT ON DONE
6135 024502 001775 BEQ      1$
6136 024504 100030 BPL      3$              ;TAKE NORMAL EXIT IF NO ERROR
6137 024506 012737 024562 001200 MOV      #2$,$ESCAPE      ;ESCAPE TO 2$ ON ERROR
6138 024514 013737 003566 001306 MOV      @#DTADPB+12,@#CYL.DS ;CYLINDER
6139 024522 113737 003565 001312 MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
6140 024530 113737 003564 001310 MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
6141 024536 012746 003572 MOV      @#DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6142 024542 004737 024026 JSR      PC,@#ERINDX     ;FORM DISPATCH INDEX
6143 024546 062607 ADD      (SP)+,PC        ;REPORT PROPER ERROR
6144 024550 104041 ERROR    41
6145 024552 104042 ERROR    42              ;PARITY ERROR
6146 024554 104043 ERROR    43              ;UNSAFE ERROR
6147 024556 104044 ERROR    44              ;NON-I/O ERROR
6148 024560 104045 ERROR    45              ;I/O ERROR
6149 024562 005720 2$: TST     (RD)+      ;ADJUST FOR ERROR EXIT
6150 024564 000404 BR        4$              ;GO TO THE EXIT
6151 024566 005064 000006 3$: CLR      RHDA(R4)     ;TRACK AND SECTOR = 0
6152 024572 005064 000034 CLR      RHCA(R4)     ;CYLINDER = 0
6153 024576 000200 4$: RTS      RD        ;RETURN

```

;SBTTL DORTI - RETURN FROM INTERRUPT
;*THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS

```

6158 024600 000002 DORTI: RTI              ;RETURN FROM INTERRUPT

```

;SBTTL STRTMR - START THE TIMERS
;*THIS ROUTINE WILL INITIALIZE THE TIMERS
;*USED BY THE "TIMING ROUTINES"
;*CALL

```

6166 * JSR      PC,@#STRTMR
6167 * RETURN
6168
6169 024602 104416 STRTMR: SAVREG          ;SAVE R0-R5
6170 024604 012700 001314 MOV      #TIM.UP,R0     ;START AT TIM.UP (MINIMUM)
6171 024610 012701 001350 MOV      #TIM.PT,R1     ;STOP AT TIM.PT
6172 024614 005020 1$: CLR      (R0)+      ;CLEAR
6173 024616 020001 CMP      R0,R1          ;DONE?
6174 024620 103775 BLO     1$              ;NO--BRANCH
6175 024622 012710 044134 MOV      #BUFFER,(R0)   ;SETUP POINTER
6176 024626 012737 077777 001314 MOV      #+CBIT15,@#TIM.UP ;SET MINIMUM TIME TO MAXIMUM
6177 024634 012737 077777 001332 MOV      #+CBIT15,@#TIM.DN ;POSITIVE NUMBER
6178 024642 104420 RESREG
6179 024644 000207 RTS      PC            ;RESTORE R0-R5

```

;SBTTL COUNT - THIS ROUTINE COUNTS THE ELAPSED TIME

6180
6181
6182

```

6183                                     ;*THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND
6184                                     ;*MAINTAIN THE MINIMUM AND MAXIMUM TIMES.
6185                                     ;*NOTE: THIS ROUTINE DESTROYS R2
6186                                     ;*CALL
6187                                     ;*      MOV      #TP,R3          ;PARAMETER POINTER
6188                                     ;*      MOV      FLAG,R5          ;FLAG=0=COUNT UP
6189                                     ;*                                     ;FLAG=-1=COUNT DOWN
6190                                     ;*      JSR      PC,@#COUNT
6191                                     ;*      RETURN
6192
6193 024646 012702 001314 COUNT: MOV      #TIM.UP,R2      ;PICKUP THE "UP" POINTER
6194 024652 005705          TST      R5              ;USE IT?
6195 024654 001402          BEQ      1$              ;YES--BRANCH
6196 024656 012702 001332          MOV      #TIM.DN,R2      ;NO--PICKUP "DOWN" POINTER
6197 024662 027722 154512 1$:    CMP      @PKC,(R2)+      ;LESS THAN PREVIOUS LOW?
6198 024666 002003          BGE      2$              ;NO--BRANCH
6199 024670 017762 154504 177776          MOV      @PKC,-2(R2)      ;YES--SAVE IT
6200 024676 027763 154476 000004 2$:    CMP      @PKC,4(R3)      ;LESS THAN THE LOW LIMIT?
6201 024704 002001          BGE      3$              ;NO--BRANCH
6202 024706 005212          INC      (R2)            ;YES--COUNT IT
6203 024710 005722          TST      (R2)+          ;ADVANCE THE POINTER
6204 024712 027722 154462          CMP      @PKC,(R2)+      ;GREATER THAN PREVIOUS HIGH?
6205 024716 003403          BLE      4$              ;NO--BRANCH
6206 024720 017762 154454 177776          MOV      @PKC,-2(R2)      ;YES--SAVE IT
6207 024726 027763 154446 000006 4$:    CMP      @PKC,6(R3)      ;GREATER THAN THE HIGH LIMIT?
6208 024734 003401          BLE      5$              ;NO--BRANCH
6209 024736 005212          INC      (R2)            ;YES--COUNT IT
6210 024740 005722          TST      (R2)+          ;ADVANCE THE POINTER
6211 024742 067722 154432          ADD      @PKC,(R2)+      ;ADD THIS COUNT TO THE TOTAL
6212 024746 005522          ADC      (R2)+
6213 024750 005212          INC      (R2)
6214 024752 022737 047304 001350          CMP      #BUFFER+<4*410.>, @#TIM.PT ;COUNT THIS READING
6215 024760 101406          BLOS     6$              ;@#TIM.PT ;SAVE THIS COUNT?
6216 024762 017777 154412 154360          MOV      @PKC,@TIM.PT    ;NO--BRANCH
6217 024770 062737 000002 001350          ADD      #2,@#TIM.PT     ;YES--WELL SAVE IT THEN
6218 024776 000207          RTS      PC              ;ADVANCE THE POINTER
6219                                     ;*RETURN
6220

```

```

6221 ;*****
6222 .SBTTL TYPTIM - TYPE TIMES
6223 ;*THIS ROUTINE IS USED TO TYPE THE MINIMUM,
6224 ;*MAXIMUM, AND AVERAGE TIMES FOR TESTS 7,10,11, AND 12.
6225 ;*IT WILL ALSO CHECK THE TIMES TO INSURE
6226 ;*THEY ARE WITHIN TOLERANCE AND IF NOT FLAG THE BAD TIMES.
6227 ;*NOTE: THIS ROUTINE DESTROYS R2-R5
6228 ;*CALL
6229 ;*      JSR      RO,@#TYPTIM      ;GO REPORT THE TIMES
6230 ;*      TABLE      ;POINT TO THE PROPER TABLE
6231 ;*      RETURN
6232 ;*TABLE:MSGADR1      ;ADDRESS OF ASCIZ MESSAGE NUMBER 1
6233 ;*      MSGADR2      ;ADDRESS OF ASCIZ MESSAGE NUMBER 2
6234 ;*      MIN.ALLOWED  ;MINIMUM TIME ALLOWED
6235 ;*      MAX.ALLOWED  ;MAXIMUM TIME ALLOWED
6236

```

```

6237 025000 012002 TYPTIM: MOV      (RO)+,R2      ;PICKUP THE TABLE POINTER
6238 025002 032737 000100 177570          BIT      #SW06,@#SWR     ;INHIBIT TIME REPORTS?

```


K10

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
 DERPKC.P11 TYPTIM - TYPE TIMES

MACY11 27(732) 14-OCT-76 10:36 PAGE 128

Address	OpCode	Op1	Op2	Op3	Op4	Comments
6239	025010	001145				BNE 7\$;YES--BRANCH
6240	025012	012237	025032			MOV (R2)+,2\$;ADDRESS OF MESSAGE NUMBER 1
6241	025016	012205				MOV (R2)+,R5 ;ADDRESS OF MESSAGE NUMBER 2
6242	025020	012203				MOV (R2)+,R3 ;PICKUP THE LOW LIMIT
6243	025022	011202				MOV (R2),R2 ;AND THE HIGH LIMIT
6244	025024	012704	001314			MOV #TIM.UP,R4 ;PARAMETER POINTER
6245	025030	104400				1\$: TYPE ;TYPE THE MESSAGE
6246	025032	000000				2\$: .WORD 0 ;ASCIZ MESSAGE POINTER GOES HERE
6247	025034	005764	000014			TST 14(R4) ;DID ANY COUNTS OCCUR?
6248	025040	001527				BEQ 6\$;NO--BRANCH
6249	025042	104400	040452			TYPE ,MSGMIN ;"MIN="
6250	025046	012446				MOV (R4)+,-(SP) ;PUT (R4)+ ON THE STACK
6251	025050	004737	021170			JSR PC,@\$\$SB2D ;CHANGE (R4)+ TO DECIMAL ASCIZ
6252	025054	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6253	025060	104400	040477			TYPE ,MSGOUS ;"0 US"
6254	025064	005724				TST (R4)+ ;ANY SEEKS BELOW THE LOW LIMIT
6255	025066	001421				BEQ 3\$;NO--BRANCH
6256	025070	104400	040612			TYPE ,MSG.SP ;
6257	025074	016446	177776			MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
6258	025100	004737	021170			JSR PC,@\$\$SB2D ;CHANGE -2(R4) TO DECIMAL ASCIZ
6259	025104	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6260	025110	104400	040504			TYPE ,M BELOW ;"BELOW THE MINIMUM OF"
6261	025114	010346				MOV R3,-(SP) ;PUT R3 ON THE STACK
6262	025116	004737	021170			JSR PC,@\$\$SB2D ;CHANGE R3 TO DECIMAL ASCIZ
6263	025122	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6264	025126	104400	040477			TYPE ,MSGOUS ;
6265	025132	104400	040461			3\$: TYPE ,MSGMAX ;"MAX="
6266	025136	012446				MOV (R4)+,-(SP) ;PUT (R4)+ ON THE STACK
6267	025140	004737	021170			JSR PC,@\$\$SB2D ;CHANGE (R4)+ TO DECIMAL ASCIZ
6268	025144	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6269	025150	104400	040477			TYPE ,MSGOUS ;
6270	025154	005724				TST (R4)+ ;ANY SEEKS ABOVE THE HIGH LIMIT
6271	025156	001421				BEQ 4\$;NO--BRANCH
6272	025160	104400	040612			TYPE ,MSG.SP ;YES--REPORT HOW MANY
6273	025164	016446	177776			MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK
6274	025170	004737	021170			JSR PC,@\$\$SB2D ;CHANGE -2(R4) TO DECIMAL ASCIZ
6275	025174	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6276	025200	104400	040533			TYPE ,M ABOVE ;"ABOVE THE MAXIMUM OF"
6277	025204	010246				MOV R2,-(SP) ;PUT R2 ON THE STACK
6278	025206	004737	021170			JSR PC,@\$\$SB2D ;CHANGE R2 TO DECIMAL ASCIZ
6279	025212	004737	021414			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6280	025216	104400	040477			TYPE ,MSGOUS ;
6281	025222	104400	040470			4\$: TYPE ,MSGAVG ;"AVG="
6282	025226	012446				MOV (R4)+,-(SP) ;FORM THE AVERAGE
6283	025230	012446				MOV (R4)+,-(SP)
6284	025232	012446				MOV (R4)+,-(SP)
6285	025234	004737	021454			JSR PC,@\$\$DIV
6286	025240	006126				ROL (SP)+ ;IS THE REMAINDER OVER HALF?
6287	025242	100001				BPL 5\$;NO--BRANCH
6288	025244	005216				INC (SP) ;YES--ROUND UP
6289	025246					5\$: JSR PC,@\$\$SB2D ;CHANGE TO DECIMAL ASCIZ
6290	025246	004737	021170			JSR PC,@\$\$SUPRS ;TYPE WITHOUT LEADING ZEROS
6291	025252	004737	021414			JSR PC,@\$\$SUPRS
6292	025256	104400	040477			TYPE ,MSGOUS
6293	025262	104400	040612			TYPE ,MSG.SP
6294	025266	016446	177776			MOV -2(R4),-(SP) ;PUT -2(R4) ON THE STACK

6295 025272 004737 021170
6296 025276 004737 021414
6297 025302 104400 040562
6298 025306 010537 025032
6299 025312 001404
6300 025314 005005
6301 025316 000644
6302 025320 104400 040577
6303 025324 000200
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314 025326 020237 002342
6315 025332 001410
6316 025334 063702 002344
6317 025340 020237 002342
6318 025344 003402
6319 025346 013702 002342
6320 025352 005720
6321 025354 000200
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333 025356 020137 002334
6334 025362 001410
6335 025364 063701 002336
6336 025370 020137 002334
6337 025374 003402
6338 025376 013701 002334
6339 025402 005720
6340 025404 000200
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350

```

JSR PC, @$$SB2D ;CHANGE -2(R4) TO DECIMAL ASCIZ
JSR PC, @$$SUPRS ;TYPE WITHOUT LEADING ZEROS
TYPE MSGNUM ;"SEEKS TIMED"
MOV R5, 2$ ;NEXT MESSAGE POINTER
BEQ 7$ ;IF NONE EXIT
CLR R5 ;NO MORE THAN 2
BR 1$
6$: TYPE MSGNON
7$: RTS R0 ;EXIT

;*****
;SBTTL INCTRK - INCREMENT TRACK NUMBER
;THIS SUBROUTINE WILL INCREMENT THE TRACK
;NUMBER (R2) BY THE AMOUNT SPECIFIED BY IT15.
;CALL
;*
;* JSR R0, @#INCTRK
;* RETURN1 ;TRACK NUMBER GREATER THAN LT15
;* RETURN2 ;TRACK NUMBER INCREMENTED
INCTRK: CMP R2, @#LT15 ;LAST TRACK COMPLETED?
        BEQ 2$ ;YES--EXIT
        ADD @#IT15, R2 ;NO--UPDATE TRACK
        CMP R2, @#LT15 ;TRACK TO BIG?
        BLE 1$ ;NO--EXIT
        MOV @#LT15, R2 ;YES--SET TRACK TO LAST TRACK
1$: TST (R0)+ ;ADJUST FOR RETURN 2
2$: RTS R0 ;RETURN

;*****
;SBTTL INCCYL - INCRMENT CYLINDER NUMBER
;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
;NUMBER (R1) BY THE AMOUNT SPECIFIED BY IC15.
;CALL
;*
;* JSR R0, @#INCCYL
;* RETURN1 ;CYLINDER NUMBER GREATER THAN LC15
;* RETURN2 ;CYLINDER NUMBER INCREMENTED
INCCYL: CMP R1, @#LC15 ;LAST CYLINDER COMPLETED?
        BEQ 2$ ;YES--EXIT
        ADD @#IC15, R1 ;NO--UPDATE CYLINDER
        CMP R1, @#LC15 ;CYLINDER TO BIG?
        BLE 1$ ;NO--EXIT
        MOV @#LC15, R1 ;YES--SET CYLINDER TO LAST CYLINDER
1$: TST (R0)+ ;ADJUST FOR RETURN 2
2$: RTS R0 ;RETURN

;*****
;SBTTL FILBUF - FILL BUFFER WITH ADDRESS DATA
;THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
;WITH ADDRESSES FROM 0 TO 21 WITH EACH ADDRESS
;BEING STORED IN 256 CONSECUTIVE LOCATIONS
;CALL
;*
;* JSR PC, @#FILBUF
;* RETURN

```


M10

6351	025406	004037	037744	FILBUF: JSR	RO, @SAVR05	;SAVE RO-R5
6352	025412	005000		CLR	RO	;FIRST DISK ADDRESS
6353	025414	012701	044134	MOV	#BUFFER, R1	;START FILLING HERE
6354	025420	012702	000400	1\$: MOV	#D256, R2	;DO 256 WORDS
6355	025424	010021		2\$: MOV	RO, (R1)+	;STORE
6356	025426	005302		DEC	R2	;MORE?
6357	025430	003375		BGT	2\$;YES--BRANCH
6358	025432	005200		INC	RO	;NO--UPDATE DISK ADDRESS
6359	025434	022700	000026	CMP	#D22, RO	;DONE?
6360	025440	003367		BGT	1\$;NO--BRANCH
6361	025442	004037	037770	JSR	RO, @GETR05	;RESTORE RO-R5
6362	025446	000207		RTS	PC	;RETURN
6363						
6364				;*****		
6365				.SBTTL CLRBUF - CLEAR BUFFER		
6366				;*THIS ROUTINE WILL CLEAR THE BUFFER BY		
6367				;*SETTING EACH WORD TO "177400".		
6368				;*CALL		
6369				* JSR	RO, @CLRBUF	
6370				* RETURN		
6371						
6372	025450	004037	037744	CLRBUF: JSR	RO, @SAVR15	;SAVE R1-R5
6373	025454	012701	177400	MOV	#177400, R1	;WORD TO FILL BUFFER WITH
6374	025460	012702	044134	MOV	#BUFFER, R2	;FIRST ADDRESS OF BUFFER
6375	025464	012703	072134	MOV	#BUFFER+<512.*22.>, R3	;LAST ADDRESS+2 OF BUFFER
6376	025470	010122		1\$: MOV	R1, (R2)+	;FILL WORDS 1, 9, ... 249, ... 5625
6377	025472	010122		MOV	R1, (R2)+	;FILL WORDS 2, 10, ... 250, ... 5626
6378	025474	010122		MOV	R1, (R2)+	;FILL WORDS 3, 11, ... 251, ... 5627
6379	025476	010122		MOV	R1, (R2)+	;FILL WORDS 4, 12, ... 252, ... 5628
6380	025500	010122		MOV	R1, (R2)+	;FILL WORDS 5, 13, ... 253, ... 5629
6381	025502	010122		MOV	R1, (R2)+	;FILL WORDS 6, 14, ... 254, ... 5630
6382	025504	010122		MOV	R1, (R2)+	;FILL WORDS 7, 15, ... 255, ... 5631
6383	025506	010122		MOV	R1, (R2)+	;FILL WORDS 8, 16, ... 256, ... 5632
6384	025510	020203		CMP	R2, R3	;DONE?
6385	025512	103766		BLO	1\$;NO--BRANCH
6386	025514	004037	037764	JSR	RO, @GETR15	;RESTORE R1-R5
6387	025520	000200		RTS	RO	;RETURN FROM CALL
6388						
6389				;*****		
6390				.SBTTL CKSCTR - CHECK SECTOR DATA		
6391				;*THIS ROUTINE IS USED TO CHECK THE DATA BUFFER		
6392				;*FOR ADDRESSES 0 THROUGH 21 WITH EACH ADDRESS		
6393				;*BEING STORED IN 256 CONSECUTIVE LOCATIONS		
6394				;*CALL		
6395				* JSR	RO, @CKSCTR	
6396				* RETURN		
6397						
6398	025522	004037	037744	CKSCTR: JSR	RO, @SAVR15	;SAVE R1-R5
6399	025526	162706	000004	SUB	#4, SP	;RESERVE TEMP. STORAGE AREA
6400	025532	005001		CLR	R1	;FIRST SECTOR
6401	025534	012716	044134	MOV	#BUFFER, (SP)	;FIRST ADDRESS OF DATA BUFFER
6402	025540	005066	000002	CLR	2(SP)	;NO ERRORS
6403	025544	012702	000020	1\$: MOV	#D16, R2	;LOOP COUNT (16*16=256)
6404	025550	011603		MOV	(SP), R3	;GET 1ST ADDRESS OF THIS SECTORS DATA
6405	025552			2\$: CMP	R1, (R3)+	;WORD 1
6406	025552	020123				

6407	025554	001064			BNE	7\$; BRANCH IF BAD
6408	025556	020123			CMP	R1, (R3)+			; WORD 2
6409	025560	001062			BNE	7\$; BRANCH IF BAD
6410	025562	020123			CMP	R1, (R3)+			; WORD 3
6411	025564	001060			BNE	7\$; BRANCH IF BAD
6412	025566	020123			CMP	R1, (R3)+			; WORD 4
6413	025570	001056			BNE	7\$; BRANCH IF BAD
6414	025572	020123			CMP	R1, (R3)+			; WORD 5
6415	025574	001054			BNE	7\$; BRANCH IF BAD
6416	025576	020123			CMP	R1, (R3)+			; WORD 6
6417	025600	001052			BNE	7\$; BRANCH IF BAD
6418	025602	020123			CMP	R1, (R3)+			; WORD 7
6419	025604	001050			BNE	7\$; BRANCH IF BAD
6420	025606	020123			CMP	R1, (R3)+			; WORD 8
6421	025610	001046			BNE	7\$; BRANCH IF BAD
6422	025612	020123			CMP	R1, (R3)+			; WORD 9
6423	025614	001044			BNE	7\$; BRANCH IF BAD
6424	025616	020123			CMP	R1, (R3)+			; WORD 10
6425	025620	001042			BNE	7\$; BRANCH IF BAD
6426	025622	020123			CMP	R1, (R3)+			; WORD 11
6427	025624	001040			BNE	7\$; BRANCH IF BAD
6428	025626	020123			CMP	R1, (R3)+			; WORD 12
6429	025630	001036			BNE	7\$; BRANCH IF BAD
6430	025632	020123			CMP	R1, (R3)+			; WORD 13
6431	025634	001034			BNE	7\$; BRANCH IF BAD
6432	025636	020123			CMP	R1, (R3)+			; WORD 14
6433	025640	001032			BNE	7\$; BRANCH IF BAD
6434	025642	020123			CMP	R1, (R3)+			; WORD 15
6435	025644	001030			BNE	7\$; BRANCH IF BAD
6436	025646	020123			CMP	R1, (R3)+			; WORD 16
6437	025650	001026			BNE	7\$; BRANCH IF BAD
6438	025652	005302			DEC	R2			; FINISHED WITH THIS SECTORS DATA?
6439	025654	001336			BNE	2\$; NO--BRANCH
6440	025656	062716	001000	3\$:	ADD	#D512, (SP)			; YES--FIRST ADDRESS OF NEXT SECTOR
6441	025662	005201			INC	R1			; MOVE TO NEXT SECTOR
6442	025664	022701	000026		CMP	#D22, R1			; DONE?
6443	025670	003325			BGT	1\$; NO--BRANCH
6444	025672	005766	000002	4\$:	TST	2(SP)			; ERROR OCCUR?
6445	025676	001406			BEQ	6\$; NO--BRANCH
6446	025700	123737	001426	001103	CMPB	Q#ERR.CT, Q#SERFLG			; MAX. ERROR OCCURRED?
6447	025706	101002			BHI	6\$; NO--BRANCH
6448	025710	013700	001270	5\$:	MOV	Q#BYPASS, R0			; TAKE ERROR EXIT
6449	025714	062706	000004	6\$:	ADD	#4, SP			; FREE TEMP. AREA
6450	025720	004037	037764		JSR	R0, Q#GETR15			; RESTORE R1-R5
6451	025724	000200			RTS	R0			; RETURN FROM CALL
6452	025726	010304		7\$:	MOV	R3, R4			; FORM WORD NUMBER AND
6453	025730	161604			SUB	(SP), R4			; ADDRESS TO CONTINUE FROM
6454	025732	010405			MOV	R4, R5			
6455	025734	006204			ASR	R4			; WORD NUMBER
6456	025736	042705	177740		BIC	#C37, R5			
6457	025742	001002			BNE	8\$; BRANCH IF NOT A MULTIPLE OF 16
6458	025744	012705	000040		MOV	#40, R5			; SET TO WORD 16
6459	025750	006305		8\$:	ASL	R5			
6460	025752	062705	025552		ADD	#2\$, R5			; ADDRESS
6461	025756	016337	177776	001126	MOV	-2(R3), Q#SBDDAT			; SAVE BAD DATA
6462	025764	005766	000002		TST	2(SP)			; FIRST ERROR?

B11

6463	025770	001015							BNE	10\$;NO--BRANCH
6464	025772	013737	003566	001306					MOV	2#DTADPB+12,2#CYL.DS	;CYLINDER NUMBER
6465	026000	113737	003565	001312					MOVB	2#DTADPB+11,2#TRK.DS	;TRACK NUMBER
6466	026006	012737	026016	001200					MOV	9\$, \$ESCAPE	;ESCAPE TO 9\$ ON ERROR
6467	026014	104021							ERROR	21	;REPORT THE ERROR
6468	026016	105166	000002		9\$:				COMB	2(SP)	;SET ERROR SWITCH
6469	026022	000404							BR	11\$	
6470	026024					10\$:					
6471	026024	012737	026034	001200					MOV	11\$, \$ESCAPE	;ESCAPE TO 11\$ ON ERROR
6472	026032	104022							ERROR	22	;REPORT THE ERROR
6473	026034	032737	001000	177570	11\$:				BIT	5W09,2#SWR	;LOOP ON ERROR?
6474	026042	001322							BNE	5\$;YES
6475	026044	032737	000002	177570					BIT	5W01,2#SWR	;STOP DATA COMPARE?
6476	026052	001307							BNE	4\$;YES--BRANCH
6477	026054	123737	001426	001103					CMPB	2#ERR.CT,2#SERFLG	;MAX. ERRORS?
6478	026062	101712							BLOS	5\$;YES--BRANCH
6479	026064	032737	000040	177570					BIT	5W05,2#SWR	;REPORT ONLY 1ST ERROR PER SECTOR?
6480	026072	001271							BNE	3\$;YES--BRANCH
6481	026074	000115							JMP	(R5)	
6482											
6483											
6484											
6485											
6486											
6487											
6488											
6489											
6490											
6491	026076	004037	037744						SETBUF: JSR	RD,2#SAVR15	;SAVE R1-R5
6492	026102	012701	044134						MOV	2#BUFFER,R1	;FIRST ADDRESS
6493	026106	122737	000017	001102					CMPB	17,\$TSTNM	
6494	026114	001002							BNE	2\$	
6495	026116	013701	001212						MOV	BUF,R1	
6496	026122	013702	003560		2\$:				MOV	2#DTADPB+4,R2	;WORD COUNT
6497	026126	016003	002434		1\$:				MOV	PAT.PT(RD),R3	;PICKUP PATTERN POINTER
6498	026132	012321							MOV	(R3)+,(R1)+	;MOVE WORD 1 INTO DATA BUFFER
6499	026134	012321							MOV	(R3)+,(R1)+	;MOVE WORD 2 INTO DATA BUFFER
6500	026136	012321							MOV	(R3)+,(R1)+	;MOVE WORD 3 INTO DATA BUFFER
6501	026140	012321							MOV	(R3)+,(R1)+	;MOVE WORD 4 INTO DATA BUFFER
6502	026142	012321							MOV	(R3)+,(R1)+	;MOVE WORD 5 INTO DATA BUFFER
6503	026144	012321							MOV	(R3)+,(R1)+	;MOVE WORD 6 INTO DATA BUFFER
6504	026146	012321							MOV	(R3)+,(R1)+	;MOVE WORD 7 INTO DATA BUFFER
6505	026150	012321							MOV	(R3)+,(R1)+	;MOVE WORD 8 INTO DATA BUFFER
6506	026152	012321							MOV	(R3)+,(R1)+	;MOVE WORD 9 INTO DATA BUFFER
6507	026154	012321							MOV	(R3)+,(R1)+	;MOVE WORD 10 INTO DATA BUFFER
6508	026156	012321							MOV	(R3)+,(R1)+	;MOVE WORD 11 INTO DATA BUFFER
6509	026160	012321							MOV	(R3)+,(R1)+	;MOVE WORD 12 INTO DATA BUFFER
6510	026162	012321							MOV	(R3)+,(R1)+	;MOVE WORD 13 INTO DATA BUFFER
6511	026164	012321							MOV	(R3)+,(R1)+	;MOVE WORD 14 INTO DATA BUFFER
6512	026166	012321							MOV	(R3)+,(R1)+	;MOVE WORD 15 INTO DATA BUFFER
6513	026170	012321							MOV	(R3)+,(R1)+	;MOVE WORD 16 INTO DATA BUFFER
6514	026172	062702	000020						ADD	1D16,R2	;DONE?
6515	026176	001353							BNE	1\$;NO--BRANCH
6516	026200	004037	037764						JSR	RD,2#GETR15	;YES--RESTORE R1-R5
6517	026204	000207							RTS	PC	;RETURN
6518											

```

*****
SBTTL SETBUF - SET BUFFER TO DATA PATTERN
*THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
*DESIRED PATTERN INTO THE DATA BUFFER.
*CALL

```

```

*      MOV      #NX,RD      ;PATTERN NUMBER INDEX TO RD
*      JSR      PC,2#SETBUF
SETBUF: JSR      RD,2#SAVR15 ;SAVE R1-R5
        MOV      #BUFFER,R1 ;FIRST ADDRESS
        CMPB    #17,$TSTNM
        BNE     2$
        MOV     BUF,R1
        MOV     2#DTADPB+4,R2 ;WORD COUNT
        MOV     PAT.PT(RD),R3 ;PICKUP PATTERN POINTER
        MOV     (R3)+,(R1)+ ;MOVE WORD 1 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 2 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 3 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 4 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 5 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 6 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 7 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 8 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 9 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 10 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 11 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 12 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 13 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 14 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 15 INTO DATA BUFFER
        MOV     (R3)+,(R1)+ ;MOVE WORD 16 INTO DATA BUFFER
        ADD     #1D16,R2 ;DONE?
        BNE     1$ ;NO--BRANCH
        JSR     RD,2#GETR15 ;YES--RESTORE R1-R5
        RTS    PC ;RETURN

```



```

6519 ;*****
6520 .SBTTL DATCMP - DATA COMPARE
6521 ;*THIS ROUTINE COMPARES A 16 WORD DATA PATTERN
6522 ;*AGAINST THE DATA BUFFER
6523 ;*CALL
6524 ;*      MOV      #NX,R0      ;PATTERN NUMBER INDEX TO RC
6525 ;*      JSR      PC,@DATCMP
6526 ;*      RETURN
6527
6528 026206 004037 037744 DATCMP: JSR      R0,@SAVR05 ;SAVE R0-R5
6529 026212 012701 044134      MOV      #BUFFER,R1 ;FIRST ADDRESS OF BUFFER
6530 026216 122737 000017 001102      CMPB     #17,STSNM
6531 026224 001002      BNE      9$
6532 026226 013701 001212      MOV      BUF,R1
6533 026232 013702 003560 9$:      MOV      @DATADPB+4,R2 ;WORD COUNT
6534 026236 005046      CLR      -(SP) ;NO ERROR
6535 026240 016003 002434 1$:      MOV      PAT.PT(R0),R3 ;PATTERN POINTER
6536 026244 2$:
6537 026244 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 1
6538 026246 001045      BNE      4$ ;BRANCH IF DIFFERENT
6539 026250 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 2
6540 026252 001043      BNE      4$ ;BRANCH IF DIFFERENT
6541 026254 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 3
6542 026256 001041      BNE      4$ ;BRANCH IF DIFFERENT
6543 026260 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 4
6544 026262 001037      BNE      4$ ;BRANCH IF DIFFERENT
6545 026264 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 5
6546 026266 001035      BNE      4$ ;BRANCH IF DIFFERENT
6547 026270 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 6
6548 026272 001033      BNE      4$ ;BRANCH IF DIFFERENT
6549 026274 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 7
6550 026276 001031      BNE      4$ ;BRANCH IF DIFFERENT
6551 026300 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 8
6552 026302 001027      BNE      4$ ;BRANCH IF DIFFERENT
6553 026304 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 9
6554 026306 001025      BNE      4$ ;BRANCH IF DIFFERENT
6555 026310 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 10
6556 026312 001023      BNE      4$ ;BRANCH IF DIFFERENT
6557 026314 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 11
6558 026316 001021      BNE      4$ ;BRANCH IF DIFFERENT
6559 026320 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 12
6560 026322 001017      BNE      4$ ;BRANCH IF DIFFERENT
6561 026324 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 13
6562 026326 001015      BNE      4$ ;BRANCH IF DIFFERENT
6563 026330 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 14
6564 026332 001013      BNE      4$ ;BRANCH IF DIFFERENT
6565 026334 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 15
6566 026336 001011      BNE      4$ ;BRANCH IF DIFFERENT
6567 026340 162321      SUB      (R3)+,(R1)+ ;CHECK WORD 16
6568 026342 001007      BNE      4$ ;BRANCH IF DIFFERENT
6569 026344 062702 000020      ADD      #1016,R2 ;DONE ?
6570 026350 001333      BNE      1$ ;NO--BRANCH
6571 026352 005726 3$:      TST      (SP)+ ;YES -- CLEAN UP STACK
6572 026354 004037 037770      JSR      R0,@GETR05 ;RESTORE R0-R5
6573 026360 000207      RTS      PC
6574 026362 010104 4$:      MOV      R1,R4 ;FORM THE WORD NUMBER

```



```

6575 026364 122737 000017 001102      CMPB  #17,$STSTNM
6576 026372 001403                      BEQ   10$
6577 026374 152704 044134              SUB   #BUFFER,R4
6578 026400 000402                      BR    11$
6579 026402 163704 001212              10$: SUB   BUF,R4
6580 026406                                11$:
6581 026406 006204                                ASR   R4 ;WORD NUMBER
6582 026410 010305                                MOV   R3,R5 ;FORM ADDRESS TO CONTINUE FROM
6583 026412 166005 002434              SUB   PAT.PT(R0),R5
6584 026416 006305                      ASL   R5
6585 026420 062705 026244              ADD   #2$,R5 ;ADDRESS
6586 026424 064341                      ADD   -(R3),-(R1) ;RECONSTRUCT THE BAD WORD
6587 026426 010137 001122              MOV   R1,@#$BDADR ;SAVE THE ERROR INFORMATION
6588 026432 010337 001120              MOV   R3,@#$GDADR
6589 026436 012137 001126              MOV   (R1)+,@#$BDDAT
6590 026442 012337 001124              MOV   (R3)+,@#$GDDAT
6591 026446 005716                      TST   (SP)
6592 026450 001023                      BNE   6$ ;1ST DATA COMPARE ERROR?
6593 026452 013737 003566 001306          MOV   @#DTADPB+12,@#CYL.DS ;CYLINDER
6594 026460 113737 003565 001312          MOVB  @#DTADPB+11,@#TRK.DS ;TRACK
6595 026466 113737 003564 001310          MOVB  @#DTADPB+10,@#SEC.DS ;SECTOR
6596 026474 016600 000016                      MOV   16(SP),R0 ;GET TEST PC+4
6597 026500 012737 026510 001200          MOV   #5$, $ESCAPE ;ESCAPE TO 5$ ON ERROR
6598 026506 104013                      ERROR 13 ;REPORT THE ERROR
6599 026510 016600 000014 5$:          MOV   14(SP),R0 ;PATTERN NUMBER INDEX
6600 026514 105116                      COMB  (SP) ;SET THE ERROR SWITCH
6601 026516 000404                      BR    7$
6602 026520                                6$:
6603 026520 012737 026530 001200          MOV   #7$, $ESCAPE ;ESCAPE TO 7$ ON ERROR
6604 026526 104014                      ERROR 14 ;REPORT THE ERROR
6605 026530 032737 000002 177570 7$:      BIT   #SW01,@#SWR ;STOP DATA COMPARE?
6606 026536 001305                      BNE   3$ ;YES--EXIT
6607 026540 123737 001426 001103          CMPB  @#ERR.CT,@#SERFLG ;MAX. ERRORS?
6608 026546 101004                      BHI   8$ ;NO--BRANCH
6609 026550 013766 001270 000016          MOV   @#B:PASS,16(SP) ;YES--ERROR EXIT
6610 026556 000675                      BR    3$
6611 026560 000115 8$:          JMP   (R5) ;NO--CONTINUE AT NEXT WORD
6612
6613 ;*****
6614 ;SBTTL FILRAN - FILL DATA BUFFER WITH RANDOM PATTERN
6615 ;*THIS ROUTINE WILL FILL THE DATA BUFFER (256*22 WORDS) WITH
6616 ;*A RANDOM PATTERN. THE FIRST TWO WORDS OF EVERY 256 WILL
6617 ;*BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
6618 ;*NEXT 254 WORDS.
6619 ;*NOTE: THIS ROUTINE DESTROYS R1 AND R2
6620 ;*CALL
6621 ;* JSR R0,@#FILRAN
6622 ;* RETURN
6623
6624 026562 012701 044134 001102  FILRAN: MOV #BUFFER,R1
6625 026566 122737 000017 001102  CMPB #17,$STSTNM
6626 026574 001002                      BNE  2$
6627 026576 013701 001212              MOV  BUF,R1
6628 026602 012702 000026 2$:          MOV  #1D22,R2
6629 026606 004037 027066 1$:          JSR  R0,@#RANPAT
6630 026612 005302                      DEC  R2

```

```

6631 026614 003374
6632 026616 000200
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643
6644 026620 013746 022022
6645 026624 013746 022024
6646 026630 012702 045134
6647 026634 012701 046134
6648 026640 122737 000017 001102
6649 026646 001007
6650 026650 013702 001212
6651 026654 010201
6652 026656 062702 001000
6653 026662 062701 002000
6654 026666 010103 7$:
6655 026670 011237 022024
6656 026674 016237 000002 022022
6657 026702 004037 027066
6658 026706 012637 022024
6659 026712 012637 022022
6660 026716 005046
6661 026720 162322 1$:
6662 026722 001452
6663 026724 012737 027020 001200
6664 026732 064342
6665 026734 010237 001122
6666 026740 010337 001120
6667 026744 012237 001126
6668 026750 012337 001124
6669 026754 010204
6670 026756 122737 000017 001102
6671 026764 001005
6672 026766 163704 001212
6673 026772 162704 001000
6674 026776 000402
6675 027000 162704 045134 9$:
6676 027004 006204 8$:
6677 027006 005716
6678 027010 001002
6679 027012 105116
6680 027014 104015
6681 027016 104016 2$:
6682 027020 032737 001000 177570 3$:
6683 027026 001012
6684 027030 123737 001426 001103
6685 027036 101406
6686 027040 032737 000002 177570

```

```

;*****
.SBTTL RANCK - RANDOM PATTERN BUFFER CHECK
;*THIS ROUTINE USES THE FIRST TWO WORDS OF THE
;*READ BUFFER TO GENERATED A RANDOM PATTERN. THEN
;*THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED.
;*NOTE: THIS ROUTINE DESTROYS R1-R4
;*CALL
;*
;* JSR RD,#RANCK
;* RETURN
RANCK: MOV #SHINUM,-(SP) ;SAVE THE PRESENT RANDOM NUMBER
MOV #SLONUM,-(SP)
MOV #BUFFER+512.,R2 ;READ BUFFER ADDRESS
MOV #BUFFER+1024.,R1 ;RANDOM PATTERN ADDRESS
CMPB #17,$STSTM
BNE 7$
MOV BUF,R2
MOV R2,R1
ADD #512.,R2
ADD #1024.,R1
7$: MOV R1,R3 ;COPY IT INTO R3 FOR LATER USE
MOV (R2),#SLONUM ;PRIME THE RANDOM NUMBER GENERATOR
MOV 2(R2),#SHINUM
JSR RD,#RANPAT ;GENERATE A RANDOM PATTERN
MOV (SP)+,#SLONUM ;RESTORE PRESENT RANDOM NUMBER
MOV (SP)+,#SHINUM
CLR -(SP) ;NO ERRORS
SUB (R3)+,(R2)+ ;ARE THESE TWO WORDS DIFFERENT?
BEQ 4$ ;NO--BRANCH
MOV #3$,$ESCAPE ;ESCAPE TO 3$ ON ERROR
ADD -(R3)-,(R2) ;RECREATE THE BAD WORD
MOV R2,#$BDADR ;ADDRESS OF BAD DATA
MOV R3,#$GDADR ;ADDRESS OF GOOD DATA
MOV (R2)+,#$BDDAT ;BAD DATA
MOV (R3)+,#$GDDAT ;GOOD DATA
MOV R2,R4 ;FORM WORD NUMBER (1 TO 256)
CMPB #17,$STSTM
BNE 9$
SUB BUF,R4
SUB #512.,R4
BR 8$
9$: SUB #BUFFER+512.,R4
8$: ASR R4
TST (SP) ;FIRST ERROR
BNE 2$ ;NO--BRANCH
COMB (SP) ;YES--SET ERROR SWITCH
ERROR 15 ;REPORT THE ERROR
ERROR 16 ;REPORT THE ERROR
2$: BIT #SW09,#SWR ;LOOP ON ERROR?
3$: BNE 5$ ;YES--BRANCH
CMPB #ERR.CT,#SERFLG ;MAX. ERRORS OCCURRED?
BLOS 5$ ;YES--BRANCH
BIT #SW01,#SWR ;STOP COMPARING?

```



```

6687 027046 001002
6688 027050 020103
6689 027052 101322
6690 027054 005726
6691 027056 001402
6692 027060 013700 001270
6693 027064 000200
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703
6704
6705 027066 010246
6706 027070 012702 000200
6707 027074 000402
6708 027076 004737 021676
6709 027102 013721 022024
6710 027106 013721 022022
6711 027112 005302
6712 027114 003370
6713 027116 012602
6714 027120 000200
6715
6716
6717
6718
6719
6720
6721
6722
6723
6724
6725 027122 004737 021676
6726 027126 113701 022024
6727 027132 113702 022025
6728 027136 013703 022022
6729 027142 105701
6730 027144 002403
6731 027146 122701 000026
6732 027152 003003
6733 027154 000241
6734 027156 106001
6735 027160 000772
6736 027162 105702
6737 027164 002403
6738 027166 122702 000023
6739 027172 003003
6740 027174 000241
6741 027176 106002
6742 027200 000772
  
```

```

      BNE      5$      ;YES--BRANCH
4$:   CMP      R1,R3  ;ALL DATA BEEN COMPARED?
      BHI      1$      ;NO--BRANCH
5$:   TST      (SP)+  ;ERROR OCCUR?
      BEQ      6$      ;NO--BRANCH
      MOV      @#BYPASS,R0 ;TAKE ERROR EXIT
6$:   RTS      R0     ;EXIT

;*****
;SBTTL RANPAT - RANDOM PATTERN GENERATOR
;THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
;PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
;OF THE PATTERN.
;CALL
;*      MOV      #ADR,R1      ;ADDRESS OF THE BUFFER
;*      JSR      R0,@#RANPAT
;*      RETURN

RANPAT: MOV      R2,-(SP)      ;SAVE R2
        MOV      #10256/2,R2 ;GENERATE 256 WORDS
        BR       2$

1$:   JSR      PC,@#SRAND      ;GENERATE A RANDOM NUMBER
2$:   MOV      @#$LONUM,(R1)+ ;PUT LOW WORD IN BUFFER
        MOV      @#$HINUM,(R1)+ ;PUT HIGH WORD IN BUFFER
        DEC      R2           ;DONE?
        BGT      1$          ;NO--BRANCH
        MOV      (SP)+,R2     ;RESTORE R2
        RTS      R0         ;EXIT

;*****
;SBTTL RANADR - RANDOM ADDRESS GENERATOR
;THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
;ADDRESSES AND SAVES THEM IN THE DPB (DTADPB+10 AND DTADPB+12).
;NOTE: THIS ROUTINE DESTROYS R1-R3
;CALL
;*      JSR      R0,@#RANADR
;*      RETURN

RANADR: JSR      PC,@#SRAND      ;GENERATE A RANDOM NUMBER
        MOV      @#$LONUM,R1    ;FORM SECTOR IN R1
        MOV      @#$LONUM+1,R2  ;FORM TRACK IN R2
        MOV      @#$HINUM,R3    ;FORM CYLINDER IN R3
        TST      R1            ;INSURE THE SECTOR IS BETWEEN 0 AND 21
        BLT      2$

1$:   CMP      #1022,R1
        BGT      3$

2$:   CLC
        RORB      R1
        BR       1$

3$:   TST      R2            ;INSURE THE TRACK IS BETWEEN 0 AND 18
        BLT      5$

4$:   CMP      #1019,R2
        BGT      6$

5$:   CLC
        RORB      R2
        BR       4$
  
```

```

6743 027202 023703 002360 6$: CMP 2#FC16,R3 ;INSURE THE CYLINDER IS BETWEEN FC AND LC
6744 027206 003413 BLE 7$
6745 027210 000241 CLC
6746 027212 006003 ROR R3
6747 027214 005503 ADC R3
6748 027216 001371 BNE 6$
6749 027220 010103 MOV R1,R3
6750 027222 000303 SWAB R3
6751 027224 060203 ADD R2,R3
6752 027226 005203 INC R3
6753 027230 003364 BGT 6$
6754 027232 005403 NEG R3
6755 027234 000762 BR 6$
6756 027236 023703 002362 7$: CMP 2#LC16,R3
6757 027242 002003 BGE 8$
6758 027244 000241 CLC
6759 027246 006003 ROR R3
6760 027250 000772 BR 7$
6761 027252 023703 002360 8$: CMP 2#FC16,R3
6762 027256 003403 BLE 9$
6763 027260 005203 INC R3
6764 027262 000303 SWAB R3
6765 027264 000764 BR 7$
6766 027266 110137 003564 9$: MOVB R1,2#DTADPB+10 ;SAVE SECTOR ADDRESS
6767 027272 110237 003565 MOVB R2,2#DTADPB+11 ;SAVE TRACK ADDRESS
6768 027276 010337 003566 MOV R3,2#DTADPB+12 ;SAVE CYLINDER ADDRESS
6769 027302 000200 RTS R0 ;RETURN

```

```

6770
6771 ;*****
6772 .SBTTL GETSWR - GET THE CONTROL SWITCH SETTINGS
6773 ;*THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
6774 ;*IF SWR<07>=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
6775 ;*SETTING IS READ AND STORED.
6776 ;*NOTE: THIS ROUTINE DESTROYS R3 AND R4
6777 ;*CALL
6778 ;* JSR PC,2#GETSWR
6779 ;* RETURN ;(C.SWR)=DESIRED CONTROL SWITCHES
6780

```

```

6781 027304 032737 000200 177570 GETSWR: BIT 2#SW07,2#SWR ;READ CONTROL SWITCHES?
6782 027312 001426 BEQ 2$ ;NO--BRANCH
6783 027314 104400 027322 TYPE .+4 ;TYPE ASCIZ STRING
6784 027320 000410 BR 64$ ;GET OVER THE ASCIZ
6785 ;;.ASCIZ <15><12>/SET SWR<07>=0/
6786 027342 64$:
6787 027342 012703 040046 1$: MOV #MSG.CS,R3 ;"CONTROL SWITCHES="
6788 027346 013704 001246 MOV 2#C.SWR,R4 ;PRESENT CONTROL SWITCH SETTINGS
6789 027352 004037 027612 JSR R0,2#GETNUM ;GET THE NEW SWITCH SETTINGS
6790 027356 000771 BR 1$ ;COMMA
6791 027360 000770 BR 1$ ;PERIOD
6792 027362 010437 001246 MOV R4,2#C.SWR ;DOUBLE PERIOD-SAVE NEW SWITCH SETTING
6793 027366 000746 BR GETSWR ;LOOP
6794 027370 000207 2$: RTS PC ;RETURN FROM CALL

```

```

6795 ;*****
6796 .SBTTL GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
6797 ;*THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
6798

```


H11

```
6799 ;*OF THE RH11/RPO4 IS SETUP TO READ THE PROPER VALUE.  
6800 ;*IT WILL ALSO READ THE ADDRESS FROM THE TTY IF  
6801 ;*REQUIRED.  
6802 ;*NOTE: THIS ROUTINE DESTROYS R0-R4  
6803 ;*CALL  
6804 ;*  
6805 ;* JSR PC, @#GETADR  
6806 ;* RETURN  
6807  
6808 027372 005737 001252 GETADR: TST @#BUSADR ; INPUT FROM TTY REQUESTED?  
6809 027376 001447 BEQ 7$ ; NO--BRANCH  
6810 027400 005037 001252 CLR @#BUSADR ; YES--CLEAR THE REQUEST FLAG  
6811 027404 012700 001362 1$: MOV #RH.ADR, R0 ; FIRST ADDRESS  
6812 027410 012703 040072 MOV #MRHCS1, R3 ; "RHCS1="  
6813 027414 011004 MOV (R0), R4 ; PRESENT RHCS1 ADDRESS  
6814 027416 004037 027612 JSR R0, @#GETNUM ; GET NEW RHCS1  
6815 027422 000402 BR 2$ ; COMMA  
6816 027424 000767 BR 1$ ; PERIOD  
6817 027426 000430 BR 5$ ; DOUBLE PERIOD  
6818 027430 010420 2$: MOV R4, (R0)+ ; SAVE NEW RHCS1  
6819 027432 012703 040103 MOV #MRHVEC, R3 ; "RHVEC="  
6820 027436 011004 MOV (R0), R4 ; PRESENT RH11 VECTOR ADDRESS  
6821 027440 004037 027612 JSR R0, @#GETNUM ; GET NEW RHVEC  
6822 027444 000402 BR 3$ ; COMMA  
6823 027446 000756 BR 1$ ; PERIOD  
6824 027450 000417 BR 5$ ; DOUBLE PERIOD  
6825 027452 010420 3$: MOV R4, (R0)+ ; SAVE NEW RHVEC  
6826 027454 012703 040114 MOV #MRHPRI, R3 ; "RHPRI="  
6827 027460 011004 MOV (R0), R4 ; PRESENT RH11 PRIORITY LEVEL  
6828 027462 006304 ASL R4 ; POSITION FOR TYPEOUT  
6829 027464 006304 ASL R4  
6830 027466 006304 ASL R4  
6831 027470 000304 SWAB R4  
6832 027472 004037 027612 JSR R0, @#GETNUM ; GET NEW RHPRI  
6833 027476 000402 BR 4$ ; COMMA  
6834 027500 000401 BR 4$ ; PERIOD  
6835 027502 000404 BR 6$ ; DOUBLE PERIOD  
6836 027504 010210 4$: MOV R2, (R0) ; SAVE NEW RHPRI  
6837 027506 000736 BR 1$ ; LOOP  
6838 027510 010410 5$: MOV R4, (R0) ; SAVE INPUT  
6839 027512 000401 BR 7$  
6840 027514 010210 6$: MOV R2, (R0) ; SAVE PRIORITY  
6841 027516 013701 000004 7$: MOV @#ERRVEC, R1 ; SAVE THE ERROR VECTOR  
6842 027522 012737 027560 000004 MOV #B$, @#ERRVEC ; SETUP FOR TRAP  
6843 027530 005777 151626 TST @RH.ADR ; CHECK FOR RH11/RPO4  
6844 027534 010137 000004 MOV R1, @#ERRVEC ; RESTORE ERROR VECTOR  
6845 027540 012700 001362 MOV #RH.ADR, R0 ; FIRST ADDRESS OF NEW PARAMETERS  
6846 027544 012701 032360 MOV #RPADR, R1 ; FIRST ADDRESS OF WHERE TO PUT THEM  
6847 027550 012021 MOV (R0)+, (R1)+ ; BUS ADDRESS  
6848 027552 012021 MOV (R0)+, (R1)+ ; VECTOR ADDRESS  
6849 027554 012021 MOV (R0)+, (R1)+ ; PRIORITY LEVEL  
6850 027556 000207 RTS PC ; RETURN  
6851 027560 010137 000004 8$: MOV R1, @#ERRVEC ; RESTORE ERROR VECTOR  
6852 027564 022626 CMP (SP)+, (SP)+ ; CLEAN OFF THE STACK  
6853 027566 104010 ERROR I0 ; REPORT THE ERROR  
6854 027570 005737 000042 TST @#42 ; IS THERE A MONITOR?
```

```

6855 027574 001703          BEQ      1$          ;NO--GO ASK FOR ADDRESS
6856 027576 005037 001256   CLR      @#DRVSEL   ;YES--NO DRIVES SELECTED
6857 027602 005037 015120   CLR      @#SEOPCT  ;NO PASSES
6858 027606 000137 014744   JMP      @#SEOP    ;GO TO END OF PROGRAM
6859
6860
6861 ;*****
6862 ;SBTTL GETNUM - ROUTINE TO GET A NUMBER
6863 ;*THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
6864 ;*INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
6865 ;*IF REQUIRED.
6866 ;*NOTE: THIS ROUTINE DESTROYS R1
6867 ;*CALL
6868 ;*      MOV      #ADR,R3          ;ADDRESS OF ASCIZ MESSAGE
6869 ;*      MOV      #NUM,R4          ;OCTAL NUMBER
6870 ;*      JSR      RO,@#GETNUM
6871 ;*      RETURN1          ;INPUT TERMINATED WITH A COMMA
6872 ;*      RETURN2          ;WITH A PERIOD
6873 ;*      RETURN3          ;WITH A DOUBLE PERIOD
6874 ;*                          ;R4=INPUT NUMBER AND
6875 ;*                          ;R2=R4*32 FOR ALL
6876 ;*                          ;THREE RETURNS
6877 027612 010337 027620   GETNUM: MOV      R3,2$          ;SAVE MESSAGE POINTER
6878 027616 104400          1$:      TYPE          ;TYPE THE MESSAGE
6879 027620 000000          2$:      .WORD      0          ;MESSAGE POINTER GOES HERE
6880 027622 010446          MOV      R4,-(SP)          ;SAVE R4 FOR TYPEOUT
6881 027624 104402          TYPOC          ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6882 027626 104414          RDLIN          ;READ AN ASCIZ STRING
6883 027630 012601          MOV      (SP)+,R1          ;ADDRESS OF FIRST CHARACTER
6884 027632 004037 031646   JSR      RO,@#CK.CHR       ;CHECK ONE CHARACTER
6885 027636 027616          1$          ;ILLEGAL CHARACTER
6886 027640 027616          1$          ;CARRIAGE RETURN
6887 027642 027652          3$:      3$          ;"/"
6888 027644 027676          7$          ;"'"
6889 027646 027704          8$          ;"'"
6890 027650 027616          1$          ;DIGIT 0-9
6891 027652
6892 027652 004037 032102   3$:      JSR      RO,@#CK.NUM   ;CHECK THE NUMBER
6893 027656 027616          1$          ;ILLEGAL INPUT
6894 027660 027672          6$          ;TERMINATED WITH A "."
6895 027662 027670          5$          ;TERMINATED WITH A ",'"
6896 027664 027666          4$          ;TERMINATED WITH A ",'"
6897 027666 005720          4$:      TST      (RO)+          ;DOUBLE PERIOD
6898 027670 005720          5$:      TST      (RO)+          ;SINGLE PERIOD
6899 027672 010204          6$:      MOV      R2,R4          ;COMMA--SAVE INPUT NUMBER
6900 027674 000414          BR       10$          ;GO TO EXIT
6901 027676 105711          7$:      TSTB     (R1)          ;TERMINATOR AFTER A COMMA?
6902 027700 001346          BNE     1$          ;NO--LOOP
6903 027702 000411          BR       10$          ;YES--EXIT
6904 027704 105711          8$:      TSTB     (R1)          ;TERMINATOR AFTER A PERIOD?
6905 027706 001406          BEQ     9$          ;YES--EXIT
6906 027710 122721 000056   CMPB    #'.,(R1)+       ;NO--DOUBLE PERIOD?
6907 027714 001340          BNE     1$          ;NO--LOOP
6908 027716 105711          TSTB    (R1)          ;YES--TERMINATOR?
6909 027720 001336          BNE     1$          ;NO--LOOP
6910 027722 005720          TST     (RO)+          ;DOUBLE PERIOD

```



```

6911 027724 005720          9$:   TST      (R0)+      ;PERIOD
6912 027726 010402          10$:  MOV      R4,R2      ;COMMA--POSITION THE
6913 027730 000302                SWAB     R2          ;NUMBER IN CASE IT
6914 027732 006202                ASR     R2          ;IS THE PRIORITY LEVEL
6915 027734 006202                ASR     R2
6916 027736 006202                ASR     R2
6917 027740 000200                RTS      R0          ;EXIT
6918
6919
6920
6921
6922
6923
6924
6925
6926 027742 004037 037744      GT.PRM: JSR     R0,@#SAVROS ;SAVE R0-R5
6927 027746 005037 001256      GT.PRI: CLR     DRVSEL     ;NO DRIVE SELECTED
6928 027752 104400 027760                TYPE    +4          ;TYPE ASCIZ STRING
6929 027756 000406                BR      64$         ;GET OVER THE ASCIZ
6930
6931
6932
6933
6934 030000 004037 031646      64$:   ;;.ASCIZ    <15><12>/DRIVE(S)=/
6935 030004 027746
6936 030006 027746
6937 030010 027746
6938 030012 027746
6939 030014 027746
6940 030016 030020
6941 030020 005301
6942 030022
6943 030022 012702 000007      1$:   RDLIN     ;READ TTY
6944 030026 004037 031722      2$:   MOV      #7,R2    ;UPPER LIMIT OF INPUT
6945 030032 027746                JSR     R0,@#CK.DIG  ;CHECK THE DIGIT(S)
6946 030034 027746                GT.PRI  ;ILLEGAL INPUT
6947 030036 030044                GT.PRI  ;INPUT TO LARGE
6948 030040 030064                3$     ;TERMINATED WITH A "."
6949 030042 030064                4$     ;TERMINATED WITH A ":'"
6950 030044 156237 032346 001256 3$:  BISB    ATABIT(R2),DRVSEL ;SET THE DRIVE SELECTED BIT
6951 030052 105741                TSTB   -(R1)        ;WAS THE LINE TERMINATED?
6952 030054 001362                BNE    2$          ;NO-GET THE NEXT DRIVE
6953 030056 005037 001260                CLR     @#TSTNMS    ;DESELECT ALL TESTS
6954 030062 000406                BR      GTTST1     ;YES--SELECT TEST
6955 030064 156237 032346 001256 4$:  BISB    ATABIT(R2),DRVSEL ;SET THE SELECTED DRIVE BITS
6956 030072 004037 037770      GT.PR2: JSR     R0,@#GETROS ;RESTORE R0-R5
6957 030076 000207                RTS      PC         ;EXIT
6958
6959
6960 030100
6961 030100 104400 030106      GTTST1: TYPE    +4          ;;TYPE ASCIZ STRING
6962 030104 000403                BR      64$         ;;GET OVER THE ASCIZ
6963
6964
6965
6966 030114
6967 030114 104414
6968 030116 012601
6969 030120 122711 000056      64$:   RDLIN     ;READ AN ASCIZ STRING
        MOV      (SP)+,R1 ;POINTER TO R1
        CMPB   #'.,(R1) ;DOUBLE PERIOD?

```

6967	030124	001007			BNE	1\$;NO--BRANCH
6968	030126	122761	000056	000001	CMPB	#',,1(R1)	
6969	030134	001003			BNE	1\$	
6970	030136	105761	000002		TSTB	2(R1)	;"CR"?
6971	030142	001753			BEQ	GT.PR2	;YES--EXIT
6972	030144	005037	001260		1\$: CLR	TSTNMS	;NO TEST SELECTED
6973	030150	005003			CLR	R3	;NO TEST TO BE OPENED
6974	030152	005004			GTTST2: CLR	R4	;NO TEST BITS SET
6975	030154	121127	000123		CMPB	(R1),#'S	;ALL SEEK TESTS?
6976	030160	001003			BNE	1\$;NO--BRANCH
6977	030162	012704	000177		MOV	#177,R4	;YES--SELECT TESTS 0-6
6978	030166	000512			BR	GTTST3	
6979	030170	121127	000124		1\$: CMPB	(R1),#'T	;ALL TIMING TESTS?
6980	030174	001003			BNE	2\$;NO--BRANCH
6981	030176	012704	003600		MOV	#3600,R4	;YES--SELECT TESTS 7-12
6982	030202	000504			BR	GTTST3	
6983	030204	121127	000101		2\$: CMPB	(R1),#'A	;ALL ADDRESSING TESTS?
6984	030210	001003			BNE	3\$;NO--BRANCH
6985	030212	012704	014000		MOV	#14000,R4	;YES--SELECT TESTS 13 & 14
6986	030216	000476			BR	GTTST3	
6987	030220	121127	000104		3\$: CMPB	(R1),#'D	;DATA TEST?
6988	030224	001003			BNE	4\$;NO--BRANCH
6989	030226	012704	020000		MOV	#20000,R4	;YES--SELECT TEST 15
6990	030232	000470			BR	GTTST3	
6991	030234	121127	000105		4\$: CMPB	(R1),#'E	;EXERCISER TEST?
6992	030240	001003			BNE	5\$;NO--BRANCH
6993	030242	012704	040000		MOV	#40000,R4	;YES--SELECT TEST 16
6994	030246	000462			BR	GTTST3	
6995	030250	004037	031572		5\$: JSR	RD,#CK.OCT	;OCTAL DIGIT?
6996	030254	000460			BR	GTTST4	;NO--BRANCH
6997	030256	010205			MOV	R2,R5	;YES--SAVE IT
6998	030260	005201			INC	R1	;MOVE TO NEXT CHARACTER
6999	030262	004037	031572		JSR	RD,#CK.OCT	;OCTAL DIGIT
7000	030266	000405			BR	6\$;NO--BRANCH
7001	030270	005201			INC	R1	;MOVE TO NEXT CHARACTER
7002	030272	006305			ASL	R5	;SCALE HIGH DIGIT
7003	030274	006305			ASL	R5	
7004	030276	006305			ASL	R5	
7005	030300	060502			ADD	R5,R2	;COMBINE HIGH & LOW DIGITS
7006	030302	020227	000016		6\$: CMP	R2,#16	;LEGAL TEST NUMBER?
7007	030306	003274			BGT	GTTST1	;NO--BRANCH
7008	030310	006302			ASL	R2	
7009	030312	016204	001430		MOV	BITS(R2),R4	;SELECT TEST
7010	030316	121127	000055		CMPB	(R1),#'-	;TEST STRING?
7011	030322	001035			BNE	GTTST4	;NO--BRANCH
7012	030324	005201			INC	R1	;YES--MOVE TO NEXT CHARACTER
7013	030326	004037	031572		JSR	RD,#CK.OCT	;OCTAL DIGIT?
7014	030332	000662			BR	GTTST1	;NO--BRANCH
7015	030334	010205			MOV	R2,R5	;YES--SAVE IT
7016	030336	005201			INC	R1	;MOVE TO NEXT CHARACTER
7017	030340	004037	031572		JSR	RD,#CK.OCT	;OCTAL DIGIT?
7018	030344	000405			BR	7\$;NO--BRANCH
7019	030346	005201			INC	R1	;YES--MOVE TO NEXT CHARACTER
7020	030350	006305			ASL	R5	;SCALE HIGH DIGIT
7021	030352	006305			ASL	R5	
7022	030354	006305			ASL	R5	


```

7023 030356 060502          ADD      R5,R2          ;COMBINE HIGH & LOW DIGIT
7024 030360 020227 000017  7$:      CMP      R2,#17      ;LEGAL TEST NUMBER?
7025 030364 003245          BGT     GTTST1        ;NO--BRANCH
7026 030366 006302          ASL     R2
7027 030370 020462 001430          CMP     R4,BITS(R2)   ;IS THE FIRST NUMBER OF THE
7028                                ;STRING SMALLER THAN THE LAST?
7029 030374 002241          BGE     GTTST1        ;NO--BRANCH
7030 030376 056204 001430  8$:      BIS     BITS(R2),R4 ;YES--SET TEST SELECT BIT FOR SELECTED TESTS
7031 030402 005742          TST     -(R2)        ;ADJUST POINTER
7032 030404 036204 001430          BIT     BITS(R2),R4 ;DONE?
7033 030410 001772          BEQ     8$           ;NO--BRANCH
7034 030412 000401          BR     GTTST4        ;SKIP THE INCREMENT
7035 030414 005201          GTTST3: INC    R1      ;MOVE TO NEXT CHARACTER
7036 030416 050437 001260  GTTST4: BIS     R4,TSTNMS ;SET SELECTED TEST BITS INTO
7037                                ;TEST SELECT WORD
7038 030422 121127 000056          CMPB   (R1),#' .    ;"PERIOD"?
7039 030426 001420          BEQ     GTTST5        ;YES--BRANCH
7040 030430 005704          TST     R4           ;ANY TEST SELECTED THIS CYCLE?
7041 030432 001622          BEQ     GTTST1        ;NO--BRANCH
7042 030434 121127 000057          CMPB   (R1),#'/    ;"OPEN"?
7043 030440 001002          BNE     1$           ;NO--BRANCH
7044 030442 050403          BIS     R4,R3       ;YES--SET BITS FOR TEST TO OPEN
7045 030444 000403          BR     2$
7046 030446 121127 000054  1$:      CMPB   (R1),#' , ;"COMMA"?
7047 030452 001212          BNE     GTTST1        ;NO--BRANCH
7048 030454 005201  2$:      INC     R1          ;MOVE TO NEXT CHARACTER
7049 030456 105711          TSTB   (R1)         ;"CR"?
7050 030460 001234          BNE     GTTST2        ;NO--GO GET NEXT CHARACTER
7051 030462 005703          TST     R3           ;ANY TESTS TO OPEN?
7052 030464 001026          BNE     OPNTST        ;YES--BRANCH
7053 030466 000604          BR     GTTST1        ;NO--START AGAIN
7054 030470 005201  GTTST5: INC    R1      ;MOVE TO NEXT CHARACTER
7055 030472 121127 000056          CMPB   (R1),#' .    ;"PERIOD"?
7056 030476 001410          BEQ     GTTST6        ;YES--BRANCH
7057 030500 105711          TSTB   (R1)         ;"CR"?
7058 030502 001402          BEQ     1$           ;YES--BRANCH
7059 030504 000137 030100          JMP     GTTST1
7060 030510 005703  1$:      TST     R3           ;ANY TEST TO BE OPENED?
7061 030512 001013          BNE     OPNTST        ;YES--BRANCH
7062 030514 000137 030072          JMP     GT.PR2
7063 030520 005201  GTTST6: INC    R1      ;MOVE TO NEXT CHARACTER
7064 030522 105711          TSTB   (R1)         ;"CR"?
7065 030524 001402          BEQ     1$           ;YES--BRANCH
7066 030526 000137 030100          JMP     GTTST1
7067 030532 005703  1$:      TST     R3           ;ANY TEST TO BE OPENED?
7068 030534 001002          BNE     OPNTST        ;YES--BRANCH
7069 030536 000137 030072          JMP     GT.PR2
7070                                ;OPEN THE TEST FOR CHANGES
7071 030542 004037 037744  OPNTST: JSR   RO,#SAVR05 ;START WITH TEST 0
7072 030546 005027          CLR    (PC)+        ;COUNT STORED HERE
7073 030550 000000          OPN.CT: .WORD 0      ;SKIP THE INCREMENT
7074 030552 000412          BR     OPN.2
7075 030554 005237 030550  OPN.1: INC    OPN.CT   ;MOVE TO THE NEXT TEST
7076 030560 022737 000016 030550  CMP     #16,OPN.CT  ;TEST NUMBER TO BIG?
7077 030566 002004          BGE    OPN.2
7078 030570 004037 037770          JSR   RO,#GETR05   ;NO--OPEN THE NEXT TEST

```

7079	030574	000137	030100		JMP	GTTST1		: YES--GO ASK FOR MORE TESTS
7080	030600	013705	030550	OPN.2:	MOV	OPN.CT, R5		: SETUP TO USE THE
7081	030604	006305			ASL	R5		: TEST NUMBER AS AN INDEX
7082	030606	036503	001430		BIT	BITS(R5), R3		: OPEN THIS TEST?
7083	030612	001760			BEQ	OPN.1		: NO--MOVE TO NEXT TEST
7084	030614	104400	030622		TYPE	+4		: TYPE ASCIZ STRING
7085	030620	000404			BR	64\$: GET OVER THE ASCIZ
7086								
7087	030632			64\$:	::.ASCIZ	/ TEST /		
7088	030632	013746	030550		MOV	OPN.CT, -(SP)		: SAVE OPN.CT FOR TYPEOUT
7089								: TEST NUMBER
7090	030636	104404			TYPOS			: GO TYPE--OCTAL ASCII
7091	030640	002			.BYTE	2		: TYPE 2 DIGIT(S)
7092	030641	000			.BYTE	0		: SUPPRESS LEADING ZEROS
7093	030642	104400	001207		TYPE	.\$CRLF		: TYPE "CR" & "LF"
7094	030646	016500	001516		MOV	PRMPT(R5), R0		: PICKUP PARAMETER POINTER
7095	030652	011046			MOV	(R0), -(SP)		: SAVE THE VARIABLE INDICATOR
7096	030654	012702	001470		MOV	#PRM, R2		: FIRST ADDRESS OF TABLE
7097	030660	000405			BR	2\$		
7098	030662	006216		1\$:	ASR	(SP)		: CHECK FOR A VARIABLE
7099	030664	103403			BCS	2\$: GO MOVE THIS ONE
7100	030666	001404			BEQ	OPNPRM		: DONE
7101	030670	005722			TST	(R2)+		: BUMP THE POINTER
7102	030672	000773			BR	1\$		
7103	030674	012022		2\$:	MOV	(R0)+, (R2)+		: MOVE THIS VARIABLE INTO THE
7104	030676	000771			BR	1\$: COMMON AREA
7105	030700	013716	001470	OPNPRM:	MOV	#PRM, (SP)		: GET THE VARIABLE INDICATOR
7106	030704	005004			CLR	R4		: ZERO THE INDEX
7107	030706	006216		1\$:	ASR	(SP)		: CHECK FOR A VARIABLE
7108	030710	103403			BCS	3\$: GO GET IT
7109	030712	001772			BEQ	OPNPRM		: OUT OF VARIABLES
7110	030714	005724		2\$:	TST	(R4)+		: UPDATE THE INDEX
7111	030716	000773			BR	1\$		
7112	030720	005764	001554	3\$:	TST	PRMLT(R4)		: IS THE MAX. MAGNITUDE NEG?
7113	030724	100463			BMI	OPNPAT		: YES--THEN IT IS THE PATTERN
7114	030726	104400	040612		TYPE	MSG.SP		: TYPE SPACES
7115	030732	016437	001600	030742	MOV	PRMMSG(R4), 4\$: TYPE THE NAME OF THIS VARIABLE
7116	030740	104400			TYPE			
7117	030742	000000		4\$:	.WORD	0		
7118	030744	104400	040044		TYPE	MSG.EQ		: TYPE "="
7119	030750	016446	001472		MOV	RPT(R4), -(SP)		: PUT RPT(R4) ON THE STACK
7120	030754	004737	021170		JSR	PC, #\$\$S2D		: CHANGE RPT(R4) TO DECIMAL ASCIZ
7121	030760	004737	021414		JSR	PC, #\$\$SUPRS		: TYPE WITHOUT LEADING ZEROS
7122	030764	104414			RDLIN			
7123	030766	012601			MOV	(SP)+, R1		: READ AN ASCIZ STRING
7124	030770	004037	031646		JSR	R0, #CK.CHR		: CHECK ONE CHARACTER
7125	030774	030720			3\$: ILLEGAL CHARACTER
7126	030776	030720			3\$: CARRIAGE RETURN
7127	031000	031044			3\$: "/"
7128	031002	031010			5\$: " "
7129	031004	031016			6\$: " "
7130	031006	030720			3\$: DIGIT 0-9
7131	031010	105711		5\$:	TSTB	(R1)		: "CR"?
7132	031012	001342			BNE	3\$: NO--STAY ON THIS VARIABLE
7133	031014	000737			BR	2\$: YES--MOVE TO NEXT VARIABLE
7134	031016	105711		6\$:	TSTB	(R1)		: IS THERE A "CR" AFTER THE PERIOD?

7135	031020	001002		BNE	64\$;NO
7136	031022	000137	031432	JMP	OPN.N2	;YES--GO CLOSE THIS TEST
7137	031026	122721	000056	64\$: CMPB	#'.,(R1)+	;DOUBLE PERIOD?
7138	031032	001332		BNE	3\$;NO--GO ASK FOR THIS VARIABLE
7139	031034	105711		TSTB	(R1)	;YES--IS A "CR" AFTER THE DOUBLE PERIOD?
7140	031036	001330		BNE	3\$;NO--ASK FOR THIS VARIABLE AGAIN
7141	031040	000137	031450	JMP	OPN.X2	;YES--CLOSE ALL TEST
7142	031044			7\$:		
7143	031044	016402	001554	MOV	PRMLT(R4),R2	;UPPER LIMIT OF INPUT
7144	031050	004037	031722	JSR	RO,@#CK.DIG	;CHECK THE DIGIT(S)
7145	031054	030720		3\$;ILLEGAL INPUT
7146	031056	030720		3\$;INPUT TO LARGE
7147	031060	031066		8\$;TERMINATED WITH A ","
7148	031062	031426		OPN.N1		;TERMINATED WITH A "."
7149	031064	031444		OPN.X1		;TERMINATED WITH A "..."
7150	031066	010264	001472	8\$: MOV	R2,RPT(R4)	;SAVE THIS VARIABLE
7151	031072	000710		BR	2\$;MOVE TO NEXT VARIABLE
7152	031074	104400	040612	OPNPAT: TYPE	,MSG.SP	;TYPE SPACES
7153	031100	104400	040040	TYPE	,MSG.PAT	;TYPE "PAT"
7154	031104	104400	040044	TYPE	,MSG.EQ	;TYPE "="
7155	031110	016446	001472	MOV	RPT(R4),-(SP)	;SAVE RPT(R4) FOR TYPEOUT
7156	031114	104402		TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
7157	031116	104414		RDLIN		;READ ASCII STRING
7158	031120	012601		MOV	(SP)+,R1	;PICKUP POINTER
7159	031122	004037	031646	JSR	RO,@#CK.CHR	;CHECK ONE CHARACTER
7160	031126	031074		OPNPAT		;ILLEGAL CHARACTER
7161	031130	031074		OPNPAT		;CARRIAGE RETURN
7162	031132	031162		2\$		"/"
7163	031134	030700		OPNPRM		":"
7164	031136	031142		1\$		":."
7165	031140	031074		OPNPAT		;DIGIT 0-9
7166	031142	105711		1\$: TSTB	(R1)	; "CR" AFTER THE PERIOD?
7167	031144	001532		BEQ	OPN.N2	;YES--GO CLOSE THIS TEST
7168	031146	122721	000056	CMPB	#'.,(R1)+	;NO--PERIOD?
7169	031152	001350		BNE	OPNPAT	;NO--LOOP
7170	031154	105711		TSTB	(R1)	; "CR" AFTER A DOUBLE PERIOD?
7171	031156	001534		BEQ	OPN.X2	;YES--GO START TESTING
7172	031160	000745		BR	OPNPAT	;NO--LOOP
7173	031162			2\$:		
7174	031162	004037	032102	JSR	RO,@#CK.NUM	;CHECK THE NUMBER
7175	031166	031074		OPNPAT		;ILLEGAL INPUT
7176	031170	031176		3\$;TERMINATED WITH A ","
7177	031172	031426		OPN.N1		;TERMINATED WITH A "."
7178	031174	031444		OPN.X1		;TERMINATED WITH A "..."
7179	031176	010264	001472	3\$: MOV	R2,RPT(R4)	;SAVE THE INPUT NUMBER
7180	031202	006002		ROR	R2	;OPEN PATTERN 0?
7181	031204	103235		BCC	OPNPRM	;NO--START AT BEGINNING OF PARAMETER TABLE
7182	031206	004037	037744	JSR	RO,@#SAVROS	;SAVE R1-R5
7183	031212	005000		OPNWDS: CLR	RO	;START WITH WORD 0
7184	031214	012704	002474	MOV	#PATO,R4	
7185	031220			1\$:		
7186	031220	104400	031226	TYPE	+4	;TYPE ASCII STRING
7187	031224	000403		BR	64\$;GET OVER THE ASCII
7188				64\$: ;;	.ASCIZ / WD/	
7189	031234					
7190	031234	010046		MOV	RO,-(SP)	;PUT RO ON THE STACK

7191	031236	004737	021170		JSR	PC, @\$\$SB20	:CHANGE RD TO DECIMAL ASCIZ
7192	031242	004737	021414		JSR	PC, @\$\$SUPRS	:TYPE WITHOUT LEADING ZEROS
7193	031246	104400	040044		TYPE	,MSG.EQ	:TYPE "="
7194	031252	011446			MOV	(R4), -(SP)	:SAVE (R4) FOR TYPEOUT
7195	031254	104402			TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
7196	031256	104414			RDLIN		:READ ASCIZ STRING
7197	031260	012601			MOV	(SP)+, R1	:PICKUP THE POINTER
7198	031262	004037	031646		JSR	RD, @CK.CHR	:CHECK ONE CHARACTER
7199	031266	031220			1\$:ILLEGAL CHARACTER
7200	031270	031220			1\$:CARRIAGE RETURN
7201	031272	031302			2\$:"/"
7202	031274	031320			4\$:"."
7203	031276	031334			5\$:"'."
7204	031300	031220			1\$:DIGIT 0-9
7205	031302			2\$:			
7206	031302	004037	032102		JSR	RD, @CK.NUM	:CHECK THE NUMBER
7207	031306	031220			1\$:ILLEGAL INPUT
7208	031310	031316			3\$:TERMINATED WITH A "."
7209	031312	031354			6\$:TERMINATED WITH A ":'."
7210	031314	031370			8\$:TERMINATED WITH A ":'."
7211	031316	010214		3\$:	MOV	R2, (R4)	:SAVE THE INPUT
7212	031320	005724		4\$:	TST	(R4)+	:MOVE TO NEXT WORD
7213	031322	005200			INC	RD	:INCREMENT THE COUNT
7214	031324	022700	000020		CMP	#16., RD	:COUNT TO LARGE?
7215	031330	003333			1\$:NO--BRANCH
7216	031332	000727			BR	OPN.WDS	:YES--BRANCH
7217	031334	105711		5\$:	TSTB	(R1)	: "CR" AFTER THE PERIOD?
7218	031336	001407			BEQ	7\$:YES--GO CLOSE THIS TEST
7219	031340	122721	000056		CMPB	#',., (R1)+	:NO--PERIOD?
7220	031344	001325			BNE	1\$:NO--BRANCH ILLEGAL INPUT STRING
7221	031346	105711			TSTB	(R1)	: "CR" AFTER THE "PERIOD-PERIOD"?
7222	031350	001410			BEQ	9\$:YES--GO START TESTING
7223	031352	000722			BR	1\$:NO--LOOP
7224	031354	010224		6\$:	MOV	R2, (R4)+	:SAVE THE INPUT
7225	031356	004737	031404	7\$:	JSR	PC, @CLSWDS	:CLOSE THE DATA PATTERN
7226	031362	004037	037770		JSR	RD, @GETR05	:GET RD-R5
7227	031366	000421			BR	OPN.N2	:MOVE TO NEXT TEST
7228	031370	010224		8\$:	MOV	R2, (R4)+	:SAVE THE INPUT
7229	031372	004737	031404	9\$:	JSR	PC, @CLSWDS	:CLOSE THE DATA PATTERN
7230	031376	004037	037770		JSR	RD, @GETR05	:GET RD-R5
7231	031402	000422			BR	OPN.X2	:START TESTING
7232	031404	012701	002474	CLSWDS:	MOV	#PATO, R1	:FIRST ADDRESS OF DATA PATTERN
7233	031410	005200		1\$:	INC	RD	:COUNT THE LAST WORD THAT WAS STORED
7234	031412	022700	000017		CMP	#15., RD	:END OF TABLE
7235	031416	002402			BLT	2\$:YES--EXIT
7236	031420	012124			MOV	(R1)+, (R4)+	:COPY
7237	031422	000772			BR	1\$:LOOP
7238	031424	000207		2\$:	RTS	PC	:RETURN
7239	031426	010264	001472	OPN.N1:	MOV	R2, RPT(R4)	:SAVE THIS VARIABLE
7240	031432	005726		OPN.N2:	TST	(SP)+	:CLEAN OFF THE STACK
7241	031434	004737	031506		JSR	PC, CLOSE	:CLOSE THIS TEST
7242	031440	000137	030554		JMP	OPN.1	:GO OPEN THE NEXT TEST
7243	031444	010264	001472	OPN.X1:	MOV	R2, RPT(R4)	:SAVE THIS VARIABLE
7244	031450	005726		OPN.X2:	TST	(SP)+	:CLEAN OFF THE STACK
7245	031452	004737	031506	1\$:	JSR	PC, CLOSE	:CLOSE THIS TEST
7246	031456	005725		2\$:	TST	(R5)+	:UPDATE THE INDEX


```

7247 031460 020527 000034          CMP      R5,#16*2      ;INDEX TO BIG?
7248 031464 002404          BLT      3$           ;NO--BRANCH
7249 031466 004037 037770          JSR      RO,@#GETR05  ;YES--RESTORE RO-R5
7250 031472 000137 030072          JMP      GT.PR2      ;GO TO EXIT
7251 031476 036503 001430          3$:     BIT      BITS(R5),R3 ;IS THIS TEST OPEN FOR CHANGE?
7252 031502 001363          BNE      1$           ;YES--GO CLOSE IT
7253 031504 000764          BR       2$           ;NO--MOVE TO NEXT TEST
7254 031506 004037 037744          CLOSE: JSR      RO,@#SAVR05 ;SAVE RO-R5
7255 031512 012700 001470          MOV      #PRM,RO     ;"FROM" ADDRESS
7256 031516 016501 001516          MOV      PRMP↑(R5),R1 ;"TO" ADDRESS
7257 031522 012002          MOV      (RO)+,R2    ;"FROM" INDICATOR
7258 031524 012103          MOV      (R1)+,R3    ;"TO" INDICATOR
7259 031526 012704 000001          MOV      #1,R4       ;TEST BIT START A "RPT"
7260 031532 030402          1$:     BIT      R4,R2    ;PARAMETER TO BE MOVED?
7261 031534 001403          BEQ      2$           ;NO--BRANCH
7262 031536 030403          BIT      R4,R3       ;A PLACE TO PUT IT?
7263 031540 001404          BEQ      3$           ;NO--BRANCH
7264 031542 011011          MOV      (RO),(R1)   ;YES--MOVE "FROM" TO "TO"
7265 031544 030403          2$:     BIT      R4,R3    ;"TO" PARAMETER?
7266 031546 001401          BEQ      3$           ;NO--BRANCH
7267 031550 005721          TST      (R1)+       ;YES--UPDATE THE POINTER
7268 031552 005720          3$:     TST      (RO)+    ;UPDATE FROM POINTER
7269 031554 006304          ASL      R4           ;POSITION THE TEST BIT
7270 031556 032704 002000          BIT      #BIT10,R4  ;DONE?
7271 031562 001763          BEQ      1$           ;NO--BRANCH
7272 031564 004037 037770          JSR      RO,@#GETR05 ;YES--RESTORE RO-R5
7273 031570 000207          RTS      PC           ;RETURN
7274                                     ;*****
7275 .SBTTL CK.OCT -- CHECK FOR OCTAL CHARACTER
7276 ;*THIS ROUTINE IS USED TO CHECK IF AN
7277 ;*ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
7278 ;*CALL
7279 ;*      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
7280 ;*      JSR      RO,@#CK.OCT  ;CHECK THE CHARACTER
7281 ;*      RETURN1     ;C. CHARACTER IS NOT BETWEEN 0-7
7282 ;*      RETURN2     ;CHARACTER IS IN R2 AS A
7283 ;*                                     ;OCTAL DIGIT
7284
7285 031572 121127 000060          CK.OCT: CMPB     (R1),#'0  ;LESS THAN ZERO?
7286 031576 103407          BLO      1$           ;YES -- BRANCH
7287 031600 121127 000067          CMPB     (R1),#'7     ;GREATER THAN SEVEN?
7288 031604 101004          BHI      1$           ;YES -- BRANCH
7289 031606 111102          MOV      (R1),R2     ;GET THE CHARACTER
7290 031610 042702 177770          BIC      #↑C7,R2     ;STRIP AWAY THE ASCII
7291 031614 005720          TST      (RO)+       ;ADJUST FOR RETURN
7292 031616 000200          1$:     RTS      RO     ;RETURN
7293
7294                                     ;*****
7295 .SBTTL CK.DEC -- CHECK FOR DECIMAL CHARACTER
7296 ;*THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
7297 ;*AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
7298 ;*CALL
7299 ;*      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
7300 ;*      JSR      RO,@#CK.DEC  ;CHECK THE CHARACTER
7301 ;*      RETURN1     ;NOT BETWEEN 0 AND 9
7302 ;*      RETURN2     ;BETWEEN 0 AND 9

```

```

7303 ;* ;R2 = DIGIT
7304
7305 031620 121127 000060 CK.DEC: CMPB (R1),#'0 ;LESS THAN ZERO?
7306 031624 103407 BLO 1$ ;YES -- BRANCH
7307 031626 121127 000071 CMPB (R1),#'9 ;GREATER THAN NINE?
7308 031632 101004 BHI 1$ ;YES -- BRANCH
7309 031634 111102 MOVB (R1),R2 ;GET THE CHARACTER
7310 031636 042702 000060 BIC #'0,R2 ;STRIP AWAY THE ASCII
7311 031642 005720 TST (R0)+ ;ADJUST FOR RETURN
7312 031644 000200 1$: RTS R0 ;RETURN
7313
7314 ;*****
7315 .SBTTL CK.CHR -- CHECK CHARACTER
7316 ;*THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
7317 ;*DETERMINE WHAT IT IS.
7318 ;*CALL
7319 ;* MOV #ADR,R1 ;ADDRESS OF ASCII CHARACTER
7320 ;* JSR R0,@#CK.CHR ;CHECK CHARACTER
7321 ;* RETURN ADR1 ;UNKNOWN CHARACTER
7322 ;* RETURN ADR2 ;CARRIAGE RETURN * (R1)=ADR+1
7323 ;* RETURN ADR3 ;SLASH * (R1)=ADR+1
7324 ;* RETURN ADR4 ;COMMA * (R1)=ADR+1
7325 ;* RETURN ADR5 ;PERIOD * (R1)=ADR+1
7326 ;* RETURN ADR6 ;DIGIT BETWEEN 0 AND 9.
7327 ;* ;R2 = DIGIT * (R1)=ADR+1
7328
7329 031646 105711 CK.CHR: TSTB (R1) ;"CARRIAGE RETURN"?
7330 031650 001420 BEQ 4$ ;YES -- BRANCH
7331 031652 121127 000057 CMPB (R1),#' / ;"SLASH"?
7332 031656 001414 BEQ 3$ ;YES -- BRANCH
7333 031660 121127 000054 CMPB (R1),#', ;"COMMA"?
7334 031664 001410 BEQ 2$ ;YES -- BRANCH
7335 031666 121127 000056 CMPB (R1),#'. ;"PERIOD"?
7336 031672 001404 BEQ 1$ ;YES -- BRANCH
7337 031674 004037 031620 JSR R0,@#CK.DEC ;"DIGIT"?
7338 031700 000406 BR 5$ ;NO -- BRANCH
7339 031702 005720 TST (R0)+ ;DIGIT BETWEEN 0-9
7340 031704 005720 1$: TST (R0)+ ;PERIOD
7341 031706 005720 2$: TST (R0)+ ;COMMA
7342 031710 005720 3$: TST (R0)+ ;SLASH
7343 031712 005720 4$: TST (R0)+ ;CARRIAGE RETURN
7344 031714 005201 INC R1 ;MOVE POINTER TO NEXT CHARACTER
7345 031716 011000 5$: MOV (R0),R0 ;UNKNOWN CHARACTER
7346 031720 000200 RTS R0 ;RETURN
7347
7348 ;*****
7349 .SBTTL CK.DIG - CHECK DIGIT
7350 ;*THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
7351 ;*CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
7352 ;*CALL
7353 ;* MOV #ADR,R1 ;ADDRESS OF ASCII STRING
7354 ;* MOV #NUM,R2 ;MAX. MAGNITUDE OF INPUT NUMBER
7355 ;* JSR R0,@#CK.DIG ;CHECK DIGITS
7356 ;* RETURN ADR1 ;ILLEGAL CHARACTER -- R2=?
7357 ;* RETURN ADR2 ;INPUT NUMBER TO LARGE -- R2=?
7358 ;* RETURN ADR3 ;"COMMA" -- R2 = NUMBER

```



```

7359          :*      RETURN  ADR4          ;"PERIOD" -- R2 = NUMBER
7360          :*      RETURN  ADR5          ;"PERIOD-PERIOD" -- R2 = NUMBER
7361
7362 031722 010446          CK.DIG: MOV    R4,-(SP)      ;SAVE R4
7363 031724 010346          MOV    R3,-(SP)      ;SAVE R3
7364 031726 010246          MOV    R2,-(SP)      ;SAVE THE MAX. SIZE ON THE STACK
7365 031730 005002          CLR    R2          ;START WITH 0
7366 031732 005003          CLR    R3
7367 031734 005004          CLR    R4
7368 031736 004037 031646  JSR    RD,@#CK.CHR      ;CHECK ONE CHARACTER
7369 031742 032066          B$          ;ILLEGAL CHARACTER
7370 031744 032066          B$          ;CARRIAGE RETURN
7371 031746 032066          B$          ;"/"
7372 031750 032066          B$          ;"."
7373 031752 032066          B$          ;" "
7374 031754 031756          I$          ;DIGIT 0-9
7375 031756 006303          1$: ASL    R3          ;*2
7376 031760 010346          MOV    R3,-(SP)      ;SAVE *2
7377 031762 006303          ASL    R3          ;*4
7378 031764 006303          ASL    R3          ;*8
7379 031766 062603          ADD   (SP)+,R3       ;(*8)+(*2)=*10.
7380 031770 060203          ADD   R2,R3         ;UPDATE THE INPUT NUMBER
7381 031772 004037 031646  JSR    RD,@#CK.CHR      ;CHECK ONE CHARACTER
7382 031776 032066          B$          ;ILLEGAL CHARACTER
7383 032000 032066          B$          ;CARRIAGE RETURN
7384 032002 032066          B$          ;"/"
7385 032004 032014          3$          ;"."
7386 032006 032012          2$          ;" "
7387 032010 031756          I$          ;DIGIT 0-9
7388 032012 005724          2$: TST   (R4)+       ;"PERIOD"
7389 032014 005724          3$: TST   (R4)+       ;"COMMA"
7390 032016 004037 031646  JSR    RD,@#CK.CHR      ;CHECK ONE CHARACTER
7391 032022 032066          B$          ;ILLEGAL CHARACTER
7392 032024 032056          6$          ;CARRIAGE RETURN
7393 032026 032066          B$          ;"/"
7394 032030 032066          B$          ;"."
7395 032032 032036          4$          ;" "
7396 032034 032046          5$          ;DIGIT 0-9
7397 032036 005724          4$: TST   (R4)+       ;"PERIOD-PERIOD"
7398 032040 105711          TSTB  (R1)          ;"CR"?
7399 032042 001405          BEQ   6$           ;YES--BRANCH
7400 032044 000410          BR    8$
7401 032046 126127 177776 000054 5$: CMPB  -2(R1),#',     ;WAS CHARACTER BEFORE THE DIGIT A COMMA?
7402 032054 001004          BNE   8$           ;NO--EXIT
7403 032056 020316          6$: CMP   R3,(SP)     ;INPUT TO LARGE?
7404 032060 101001          BHI   7$           ;YES -- BRANCH
7405 032062 060400          ADD   R4,R0        ;ADJUST RETURN ADDRESS
7406 032064 005720          7$: TST   (R0)+       ;NUMBER TO R2
7407 032066 010302          8$: MOV   R3,R2      ;CLEAN MAX. SIZE OFF OF STACK
7408 032070 005726          TST   (SP)+        ;RESTORE R3
7409 032072 012603          MOV   (SP)+,R3     ;RESTORE R4
7410 032074 012604          MOV   (SP)+,R4     ;GET RETURN ADDRESS
7411 032076 011000          MOV   (R0),R0      ;RETURN
7412 032100 000200          RTS   R0
7413
7414
;*****

```

```

7415 .SBTTL CK.NUM - CHECK NUMBER (OCTAL)
7416 *THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
7417 *AND FORMS AN OCTAL NUMBER IN R2
7418 *CALL:
7419 *      MOV      #ADR,R1      ;ADDRESS OF ASCIZ STRING
7420 *      JSR      RD,R#CK.NUM  ;GO FORM THE NUMBER
7421 *      RETURN   ADR1        ;ILLEGAL CHARACTER IN THE INPUT STRING
7422 *      RETURN   ADR2        ;"COMMA"--R2=NUMBER
7423 *      RETURN   ADR3        ;"PERIOD"--R2=NUMBER
7424 *      RETURN   ADR4        ;"PERIOD-PERIOD"--R2=NUMBER
7425
7426 CK.NUM: MOV      R3,-(SP)    ;SAVE R3
7427          CLR      R3        ;START NUMBER AT ZERO
7428          JSR      RD,R#CK.OCT ;OCTAL DIGIT?
7429          BR      6$        ;NO--BRANCH
7430          1$: INC     R1        ;MOVE TO NEXT CHARACTER
7431          ASL     R3        ;FOR THE OCTAL NUMBER IN R3
7432          BCS    6$        ;DON'T LET IT GET TO BIG
7433          ASL     R3
7434          BCS    6$
7435          ASL     R3
7436          BCS    6$
7437          ADD     R2,R3
7438          JSR      RD,R#CK.OCT ;IS THIS AN OCTAL DIGIT?
7439          BR      2$        ;NO--FIND OUT WHAT IT IS
7440          BR      1$        ;YES--MAKE IT PART OF THE NUMBER
7441          2$: MOV     R3,R2    ;SAVE THE OCTAL NUMBER
7442          CLR     R3        ;START WITH ZERO INDEX
7443          JSR      RD,R#CK.CHR ;CHECK ONE CHARACTER
7444          6$          ;ILLEGAL CHARACTER
7445          6$          ;CARRIAGE RETURN
7446          6$          ;"/"
7447          5$          ;" "
7448          3$          ;"'"
7449          6$          ;DIGIT 0-9
7450          3$: TST     (R3)+    ;"PERIOD"
7451          CMPB   (R1),#'.'    ;"PERIOD-PERIOD"?
7452          BNE   5$          ;NO--BRANCH
7453          INC     R1        ;YES--ADVANCE THE POINTER
7454          4$: TST     (R3)+    ;"PERIOD-PERIOD"
7455          5$: TST     (R1)+    ;"COMMA"
7456          TSTB   (R1)        ;"CR"?
7457          BNE   6$          ;NO--BRANCH
7458          ADD     R3,R0      ;YES--SAVE THE OCTAL NUMBER
7459          6$: MOV     (SP)+,R3 ;RESTORE R3
7460          MOV     (R0),R0    ;PICKUP EXIT ADDRESS
7461          RTS     R0        ;RETURN
  
```



```

7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474
7475
7476
7477
7478 032222 000000
7479 032224 000000
7480 032226 000000
7481 032230 000000
7482
7483
7484
7485
7486
7487
7488 032232 000
7489 032233 000
7490 032234 000
7491 032235 000
7492 032236 000
7493 032237 000
7494 032240 000
7495 032241 000
7496
7497
7498
7499
7500
7501
7502 032242 000
7503 032243 000
7504 032244 000
7505 032245 000
7506 032246 000
7507 032247 000
7508 032250 000
7509 032251 000
7510
7511
7512
7513
7514
7515 032252 000000
7516 032254 000000
7517 032256 000000

```

.SBTTL RH11/RP04 DRIVER (REV. 0.9)

```

;*COPYRIGHT (C) 1974
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MA 01754
;*AUTHOR: JIM LACEY

```

;*STORAGE FOR RHDS1, RHER1, RHER2, AND RHER3 ON AN ERROR "2" OR "5"

```

;*RPERRS = RHDS1
;*RPERRS+2 = RHER1
;*RPERRS+4 = RHER2
;*RPERRS+6 = RHER3

```

```

RPERRS: .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

;*TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)

```

;*DRVACT=0 IMPLIES DRIVE IS IDLE
;*DRVACT>0 IMPLIES DRIVE IS ACTIVE WITH A COMMAND
;*DRVACT<0 IMPLIES DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

```

```

DRVACT: .BYTE 0           ;DRIVE 0
        .BYTE 0           ;DRIVE 1
        .BYTE 0           ;DRIVE 2
        .BYTE 0           ;DRIVE 3
        .BYTE 0           ;DRIVE 4
        .BYTE 0           ;DRIVE 5
        .BYTE 0           ;DRIVE 6
        .BYTE 0           ;DRIVE 7

```

;*TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)

```

;*DRVSTA=0 IMPLIES DRIVE IS OFFLINE
;*DRVSTA>0 IMPLIES DRIVE IS ONLINE
;*DRVSTA<0 IMPLIES DRIVE IS UNSAFE OR NONEXISTENT

```

```

DRVSTA: .BYTE 0           ;DRIVE 0
        .BYTE 0           ;DRIVE 1
        .BYTE 0           ;DRIVE 2
        .BYTE 0           ;DRIVE 3
        .BYTE 0           ;DRIVE 4
        .BYTE 0           ;DRIVE 5
        .BYTE 0           ;DRIVE 6
        .BYTE 0           ;DRIVE 7

```

;*TABLE OF DRIVE TYPES (DRV TYP=8 WORDS)

```

;*DRV TYP WILL CONTAIN THE DRIVE TYPE OF ALL ONLINE, OFFLINE, AND
;*UNSAFE DRIVES. IF A DRIVE IS NONEXISTENT DRV TYP WILL BE ZERO.

```

```

DRV TYP: .WORD 0           ;DRIVE 0
         .WORD 0           ;DRIVE 1
         .WORD 0           ;DRIVE 2

```

```

7518 032260 000000      .WORD 0      ;DRIVE 3
7519 032262 000000      .WORD 0      ;DRIVE 4
7520 032264 000000      .WORD 0      ;DRIVE 5
7521 032266 000000      .WORD 0      ;DRIVE 6
7522 032270 000000      .WORD 0      ;DRIVE 7
7523
7524      ;*TRANSFER WAIT FLAG (TRNSWT=1 WORD)
7525      ;*THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
7526      ;*"DPB" OF THE I/O OPERATION.
7527
7528 032272 000000      TRNSWT: .WORD 0
7529
7530      ;*SEARCH WAIT KEYS (SRCHWT=1 WORD)
7531      ;*THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
7532      ;*THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
7533      ;*REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
7534      ;*EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
7535
7536 032274 000000      SRCHWT: .WORD 0
7537
7538      ;*RP04 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
7539      ;*ACTDRV=0 IMPLIES DRIVER IS INACTIVE
7540      ;*ACTDRV>0 IMPLIES DRIVER IS ACTIVE
7541
7542 032276 000      ACTDRV: .BYTE 0
7543
7544      ;*SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
7545      ;*ACTSTR=0 IMPLIES SOFTWARE TIMER ROUTINE IS INACTIVE
7546      ;*ACTSTR>0 IMPLIES SOFTWARE TIMER ROUTINE IS ACTIVE
7547
7548 032277 000      ACTSTR: .BYTE 0
7549
7550      ;*UNLOAD FLAG (ULDFLG=8 BYTES)
7551      ;*ULDFLG=0 IMPLIES NO UNLOAD COMMAND
7552      ;*ULDFLG>0 IMPLIES UNLOAD COMMAND IN PROGRESS
7553      ;*ULDFLG<0 IMPLIES UNLOAD COMMAND IN WAIT QUEUE
7554
7555 032300 000      ULDFLG: .BYTE 0      ;DRIVE 0
7556 032301 000      .BYTE 0      ;DRIVE 1
7557 032302 000      .BYTE 0      ;DRIVE 2
7558 032303 000      .BYTE 0      ;DRIVE 3
7559 032304 000      .BYTE 0      ;DRIVE 4
7560 032305 000      .BYTE 0      ;DRIVE 5
7561 032306 000      .BYTE 0      ;DRIVE 6
7562 032307 000      .BYTE 0      ;DRIVE 7
7563
7564      ;*LOOK AHEAD COUNT (LACNT=8 BYTES)
7565      ;*LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
7566
7567 032310 000      LACNT: .BYTE 0      ;DRIVE 0
7568 032311 000      .BYTE 0      ;DRIVE 1
7569 032312 000      .BYTE 0      ;DRIVE 2
7570 032313 000      .BYTE 0      ;DRIVE 3
7571 032314 000      .BYTE 0      ;DRIVE 4
7572 032315 000      .BYTE 0      ;DRIVE 5
7573 032316 000      .BYTE 0      ;DRIVE 6

```



```

7574 032317 000 .BYTE 0 ;DRIVE 7
7575
7576 ;*SAVE REGISTERS FLAG (SAVEFG =1 WORD)
7577 ;*SAVEFG <0 IMPLIES SAVE THE RH11/RP04 REGISTERS WHEN THE
7578 ;*OPERATION IS COMPLETED AS PER (DPB+14).
7579 ;*SAVEFG=0 IMPLIES SAVE THE RH11/RP04 REGISTERS, AS PER
7580 ;*(DPB+14), AFTER AN ERROR.
7581
7582 032320 000000 SAVEFG: .WORD 0
7583
7584 ;*SEEK FLAG (SEEKFG=1 WORD)
7585 ;*SEEKFG=0 IMPLIES WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
7586 ;*FOR A DATA TRANSFER START A SEARCH COMMAND
7587 ;*SEEKFG<0 IMPLIES DATA TRANSFER WILL DO IMPLIED SEEKS,
7588 ;*DISREGARD THE WINDOW
7589
7590 032322 000000 SEEKFG: .WORD 0
7591
7592 ;*TIMEOUT TABLE (TIMER=8 WORDS)
7593 ;*THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
7594
7595 032324 177777 TIMER: .WORD -1 ;DRIVE 0
7596 032326 177777 .WORD -1 ;DRIVE 1
7597 032330 177777 .WORD -1 ;DRIVE 2
7598 032332 177777 .WORD -1 ;DRIVE 3
7599 032334 177777 .WORD -1 ;DRIVE 4
7600 032336 177777 .WORD -1 ;DRIVE 5
7601 032340 177777 .WORD -1 ;DRIVE 6
7602 032342 177777 .WORD -1 ;DRIVE 7
7603
7604 ;*DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
7605 ;*DTUW<0 IMPLIES NO DATA TRANSFER UNDERWAY
7606 ;*DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
7607
7608 032344 177777 DTUW: .WORD -1
7609
7610 ;*ATTENTION BITS TABLE (ATABIT=8 BYTES)
7611 ;*THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
7612 ;*ATTENTION BIT
7613
7614 032346 001 ATABIT: .BYTE 1 ;DRIVE 0
7615 032347 002 .BYTE 2 ;DRIVE 1
7616 032350 004 .BYTE 4 ;DRIVE 2
7617 032351 010 .BYTE 10 ;DRIVE 3
7618 032352 020 .BYTE 20 ;DRIVE 4
7619 032353 040 .BYTE 40 ;DRIVE 5
7620 032354 100 .BYTE 100 ;DRIVE 6
7621 032355 200 .BYTE 200 ;DRIVE 7
7622
7623 ;*RP04 TO RH11 "MASS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
7624 ;*CALLING IT FATAL (MCPEMX=1 WORD)
7625
7626 032356 000003 MCPEMX: .WORD 3
7627
7628 ;*STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RP04),
7629 ;*RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).

```

7630 032360 176700
 7631 032362 000254 000240
 7632
 7633
 7634 032366 000004
 7635
 7636 032370 001000
 7637
 7638 032372 000200
 7639
 7640 032374 000005
 7641
 7642
 7643
 7644 000000
 7645 000002
 7646 000004
 7647 000006
 7648 000010
 7649 000012
 7650 000014
 7651 000016
 7652 000020
 7653 000022
 7654 000024
 7655 000026
 7656 000030
 7657 000032
 7658 000034
 7659 000036
 7660 000040
 7661 000042
 7662 000044
 7663 000046
 7664 000050
 7665 000052
 7666
 7667
 7668
 7669
 7670
 7671
 7672
 7673
 7674
 7675
 7676
 7677
 7678 032376 013746 177776
 7679 032402 013737 032364 177776
 7680 032410 004037 037744
 7681 032414 004737 037502
 7682 032420 012701 032222
 7683 032424 012702 032322
 7684 032430 005021
 7685 032432 020102

RPADR: .WORD 176700
 RPVEC: .WORD 254,5*32.
 ;*MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
 MXLACT: .WORD 4
 ;*MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
 MXDLTA: .WORD 8.*64.
 ;*MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
 MNDLTA: .WORD 2*64.
 ;*MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
 MXWNDW: .WORD 5

;*DEFINITIONS OF THE RH11/RP04 ADDRESS INDEXES

RHCS1=0 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
 RHWC=2 ;WORD COUNT REGISTER (NOT A DRIVE REG)
 RHBA=4 ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
 RHDA=6 ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
 RHCS2=10 ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
 RHDS1=12 ;DRIVE STATUS REGISTER (DRIVE REG 01)
 RHER1=14 ;ERROR REGISTER #1 (DRIVE REG. 02)
 RHAS=16 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
 RHLA=20 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
 RHDB=22 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
 RHMR=24 ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
 RHDT=26 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
 RHSN=30 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
 RHOF=32 ;OFFSET REGISTER (DRIVE REG. 11)
 RHCA=34 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
 RHCC=36 ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
 RHER2=40 ;ERROR REGISTER #2 (DRIVE REG. 14)
 RHER3=42 ;ERROR REGISTER #3 (DRIVE REG. 15)
 RHEC1=44 ;ECC POSITION REGISTER (DRIVE REG. 16)
 RHEC2=46 ;ECC PATTERN REGISTER (DRIVE REG. 17)
 RHBAE=50
 RHCS3=52

;*RH11/RP04 DRIVER INIT. CODE

;*THIS ROUTINE WILL DETERMINE WHICH RP04 DRIVES ARE
 ;*AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
 ;*TO THE PROPER STATE FOR EACH DRIVE.
 ;*NOTE: THIS ROUTINE CALLS DRVINT

;*CALL

* JSR PC,RPINIT
 * RETURN

RPINIT: MOV @#PS, -(SP) ;SAVE PROCESSOR STATUS
 MOV RPVEC+2, @#PS ;SET PROCESSOR STATUS TO RH11/RP04 LEVEL
 JSR RD,SAVR15 ;GO SAVE R1-R5
 JSR PC,CLRQUE ;CLEAR ALL REQUEST QUEUES
 MOV #RPERRS,R1 ;FIRST ADDRESS TO BE CLEARED
 MOV #SEEKFG,R2 ;LAST ADDRESS TO BE CLEARED
 1\$: CLR (R1)+ ;CLEAR
 CMP R1,R2 ;ARE WE DONE?


```

7686 032434 101775          BLOS 1$          ;BRANCH IF NO
7687 032436 012702 032344    MOV  #DTUW,R2    ;LAST ADDRESS
7688 032442 012721 177777    2$: MOV  #-1,(R1)+ ;INITIALIZE
7689 032446 020102          CMP  R1,R2      ;DONE?
7690 032450 101774          BLOS 2$          ;LOOP IF NO
7691 032452 005001          CLR  R1          ;DRIVE NUMBER WILL BE IN R1
7692 032454 013704 032360    MOV  RPADR,R4   ;FIRST ADDRESS OF RH11/RP04
7693 032460 012764 000040 000010  MOV  #BIT05,RHCS2(R4) ;MASSBUS INIT.
7694 032466 004037 032540    3$: JSR  RD,DRVINT ;GO INIT. A DRIVE
7695 032472 000417          BR   7$          ;
7696 032474 005201          4$: INC  R1          ;MOVE TO THE NEXT DRIVE
7697 032476 042701 177770    BIC  #↑C7,R1    ;DON'T LET THE DRIVE NUMBER GET TO BIG
7698 032502 001371          BNE  3$          ;BRANCH IF MORE DRIVES TO INIT.
7699 032504 013703 032362    MOV  RPVEC,R3   ;SETUP THE RH11/RP04 VECTOR
7700 032510 012723 034766    MOV  #ISR,(R3)+ ;
7701 032514 013713 032364    MOV  RPVEC+2,(R3) ;
7702 032520 004037 037764    JSR  RD,GETR15  ;RESTORE R1-R5
7703 032524 012637 177776    MOV  (SP)+,D#PS ;RESTORE THE PROCESSOR STATUS
7704 032530 000207          RTS  PC          ;BYE-BYE
7705 032532 105061 032242    7$: CLRB DRVSTA(R1) ;SET THE DRIVE STATUS TO OFFLINE
7706 032536 000756          BR   4$          ;GO DO THE NEXT DRIVE
7707
7708 ;*DRIVE INIT. ROUTINE
7709 ;*THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
7710 ;*AN RPO4. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
7711 ;*IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
7712 ;*INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
7713 ;*DRVSTA IS SET TO THE PROPER CONDITION.
7714
7715 ;*CALL
7716 ;*
7717 ;* MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
7718 ;* MOV RPADR,R4 ;UNIBUS ADDRESS OF RH11/RP04 (RHCS1)
7719 ;* JSR RD,DRVINT ;CALLED BY A JSR
7720 ;* RETURN1 ;ERROR OCCURRED (PARITY)
7721 ;* RETURN2 ;NORMAL RETURN
7722
7721 032540 004037 037744    DRVINT: JSR  RD,SAVR15 ;SAVE R1-R5
7722 032544 010164 000010    MOV  R1,RHCS2(R4) ;SELECT A DRIVE
7723 032550 010103          MOV  R1,R3      ;COPY THE DRIVE NUMBER INTO
7724 032552 006303          ASL  R3          ;R3 AND POSITION IT FOR TABLE INDEXING
7725 032554 112761 177777 032242  MOVVB #-1,DRVSTA(R1) ;START DRIVE STATUS AS NONEXISTENT
7726 032562 005063 032252    CLR  DRVTP(R3)  ;CLEAR THE DRIVE TYPE INDICATOR
7727 032566 112777 000111 177564  MOVVB #111,RPADR ;DO A "DRIVE CLEAR" COMMAND
7728 032574 032764 010000 000010  BIT  #BIT12,RHCS2(R4) ;NONEXISTENT DRIVE?
7729 032602 001403          BEQ  1$          ;NO---BRANCH
7730 032604 004737 037136    JSR  PC,SET.IE ;GO SET "IE" WITHOUT A "TRE"
7731 032610 000462          BR   6$          ;LEAVE THIS ROUTINE
7732 032612 004037 036544    1$: JSR  RD,RD.RP  ;READ THE DRIVE TYPE REG.
7733 032616 000026          RHD  7$          ;
7734 032620 032760          7$:
7735 032622 012605          MOV  (SP)+,R5   ;ERROR RETURN ADDRESS
7736 032624 010563 032252    MOV  R5,DRVTP(R3) ;PUT DRIVE TYPE IN R5
7737 032630 022705 020020    CMP  #20020,R5  ;SAVE THE DRIVE TYPE
7738 032634 001403          BEQ  2$          ;IS IT A SINGLE PORT RPO4?
7739 032636 022705 024020    CMP  #24020,R5  ;BRANCH IF YES
7740 032642 001043          BNE  5$          ;IS IT A DUAL PORT RPO4?
7741 032644 012746 000121    2$: MOV  #121,-(SP) ;BRANCH IF NO
;DO A "READ-IN PRESET"

```

```

7742 032650 004037 036702      JSR      RO,WRT.RP
7743 032654 000000              RHCS1
7744 032656 032760              7$
7745 032660 012746 010000      MOV      #BIT12,-(SP)      ;SET FMT22=1
7746 032664 004037 036702      JSR      RO,WRT.RP
7747 032670 000032              RHOF
7748 032672 032760              7$
7749 032674 004037 036544      JSR      RO,RO.RP      ;READ RHDS1
7750 032700 000012              RHDS1
7751 032702 032760              7$
7752 032704 012605              MOV      (SP)+,R5      ;AND SAVE IT IN R5
7753 032706 100011              BPL      4$      ;BRANCH IF ATA=0
7754 032710 116164 032346 000016  MOVB     ATABIT(R1),RHAS(R4) ;CLEAR ATTENTION BIT
7755 032716 004037 036544      JSR      RO,RO.RP      ;FIND OUT WHY ATA=1
7756 032722 000014              RHER1
7757 032724 032760              7$
7758 032726 006126              ROL      (SP)+      ;IS IT UNSAFE?
7759 032730 100412              BMI      6$      ;BRANCH IF YES
7760 032732 005105              4$:     COM      R5      ;CHECK MOL, DPR, DRY, AND VV
7761 032734 042705 167077      BIC      #1<BIT12:BIT08:BIT07:BIT06>,R5
7762 032740 001004              BNE      5$      ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
7763 032742 112761 000001 032242  MOVB     #1,DRVSTA(R1) ;SET DRIVE STATUS TO ONLINE
7764 032750 000402              BR
7765 032752 105061 032242      5$:     CLRB     DRVSTA(R1) ;DRIVE STATUS = OFFLINE
7766 032756 005720      6$:     TST      (RO)+      ;STEP OVER THE ERROR RETURN
7767 032760 004037 037764      7$:     JSR      RO,GETR15 ;RESTORE R1-R5
7768 032764 000200              RTS      RO      ;RETURN
7769
7770      ;*REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
7771
7772      ;*CALL
7773
7774      ;*      JSR      RO,@#RP04      ;CALL THE RP04 DRIVER
7775      ;*      PNTADR     ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
7776      ;*      RETURN1    ;RETURN HERE IF QUEUE IS FULL
7777      ;*      RETURN2    ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
7778      ;*      ;IS AN ERROR CONDITION
7779
7780 032766 013746 177776      RP04:   MOV      @#PS,-(SP)      ;SAVE THE CALLING STATUS
7781 032772 013737 032364 177776  MOV      RPVEC+2,@#PS    ;DON'T ALLOW ANY RP04 INTERRUPTS
7782 033000 112737 000001 032276  MOVB     #1,ACTDRV      ;SET "ACTIVE DRIVER" FLAG
7783 033006 004037 037744      JSR      RO,SAVR15      ;SAVE R1-R5
7784 033012 012002              MOV      (RO)+,R2      ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
7785 033014 005062 000016      CLR      16(R2)      ;CLEAR THE STATUS/ERROR INDICATOR
7786 033020 111201              MOVB     (R2),R1      ;PICKUP THE DRIVE NUMBER
7787 033022 105761 032242      TSTB     DRVSTA(R1)    ;CHECK DRIVES STATUS
7788 033026 003017              BGT      1$      ;BRANCH IF ONLINE
7789 033030 105761 032300      TSTB     ULDFLG(R1)    ;UNLOAD COMMAND IN QUEUE?
7790 033034 001034              BNE      3$      ;BRANCH IF YES
7791 033036 013704 032360      MOV      RPADR,R4      ;UNIBUS ADDRESS OF RHCS1
7792 033042 004037 032540      JSR      RO,DRVINT      ;GO INIT. THE DRIVE
7793 033046 000442              BR      6$      ;ERROR RETURN
7794 033050 105761 032242      TSTB     DRVSTA(R1)    ;IS DRIVE STATUS ONLINE?
7795 033054 003004              BGT      1$      ;BRANCH IF YES
7796 033056 052762 140000 000016  BIS      #BIT15:BIT14,16(R2) ;ERROR--DRIVE IS OFFLINE OR NONEXISTENT
7797 033064 000423              BR      4$      ;GO TO EXIT

```



```

7798 033066 004037 037604      1$: JSR   RO,DRVQUE      ;PUT THIS REQUEST IN QUEUE
7799 033072 000421              BR     5$             ;QUEUE IS FULL
7800 033074 122762 000103 000002  CMPB  #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
7801 033102 001003              BNE   2$             ;BR IF NO
7802 033104 112761 177777 032300  MOVB  #-1,ULDFLG(R1) ;SET THE "UNLOAD IN QUEUE" FLAG
7803 033112 105761 032232      2$: TSTB  DRVACT(R1)     ;IS THIS DRIVE ACTIVE?
7804 033116 001006              BNE   4$             ;BR IS YES
7805 033120 004737 033162      JSR   PC,OPT         ;CALL THE OPTIMIZER
7806 033124 000403              BR     4$
7807 033126 052762 120000 000016  3$: BIS  #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
7808 033134 005720              4$: TST  (R0)+
7809 033136 004037 037764      5$: JSR   RO,GETR15     ;RESTORE R1-R5
7810 033142 105037 032276      CLRB  ACTDRV        ;CLEAR "ACTIVE DRIVER" FLAG
7811 033146 012637 177776      MOV   (SP)+,2#PS    ;RETURN "PS" TO USER LEVEL
7812 033152 000200              RTS   RO             ;RETURN TO CALLER
7813 033154 004737 034402      6$: JSR   PC,C17       ;GO HANDLE THE PARITY ERROR
7814 033160 000765              BR     4$
7815
7816 ;*OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
7817 ;*CALL
7818 ;*
7819 ;*   MOV   #DRVNUM,R1   ;DRIVE NUMBER TO R1
7820 ;*   JSR   PC,OPT       ;SETUP A COMMAND
7821 ;*
7822 033162 004037 037744      OPT: JSR   RO,SAVR15     ;SAVE R1-R5
7823 033166 013746 177776      MOV   2#PS,-(SP)    ;SAVE PROC. STATUS
7824 033172 146137 032346 032274  BICB  ATABIT(R1),SRCHWT ;CLEAR "SEARCH WAIT" KEY
7825 033200 004737 037660      JSR   PC,GETREQ     ;GET "DPB" POINTER OF REQUEST
7826 033204 005702              TST   R2            ;IS THERE A REQUEST IN QUEUE?
7827 033206 001444              BEQ   6$            ;NO--BRANCH TO EXIT
7828 033210 105761 032242      TSTB  DRVSTA(R1)    ;IS DRIVE ONLINE?
7829 033214 003006              BGT   1$            ;YES--BRANCH
7830 033216 004737 037702      JSR   PC,POPQUE     ;NO--REMOVE REQUEST FROM QUEUE
7831 033222 052762 140000 000016  7$: BIS  #BIT15!BIT14,16(R2) ;SET ERROR BIT OF STATUS/ERROR INDICATOR
7832 033230 000426              BR     5$            ;BRANCH TO EXIT
7833 033232 122762 000150 000002  1$: CMPB  #150,2(R2)   ;IS THE REQUEST FOR I/O?
7834 033240 002403              BLT   2$            ;YES--BRANCH
7835 033242 004737 033760      JSR   PC,C14        ;CALL THE COMMAND INITIATOR
7836 033246 000417              BR     5$            ;BRANCH TO EXIT
7837 033250 005737 032344      2$: TST  DTUW          ;DATA TRANSFER UNDERWAY?
7838 033254 002012              BGE   4$            ;YES--GO START A SEARCH
7839 033256 005737 032322      TST  SEEKFG         ;DO IMPLIED SEEKS?
7840 033262 100404              BMI   3$            ;YES---BRANCH
7841 033264 004037 034630      JSR   RO,LA         ;NO--DO LOOK AHEAD
7842 033270 000406              BR     5$            ;RETURN HERE ON A PARITY ERROR
7843 033272 000403              BR     4$            ;GO START A SEARCH
7844 033274 004737 033326      3$: JSR   PC,C11        ;START A DATA TRANSFER
7845 033300 000402              BR     5$
7846 033302 004737 033652      4$: JSR   PC,C13        ;START A SEARCH
7847 033306 012637 177776      5$: MOV   (SP)+,2#PS    ;RESTORE PROC. STATUS
7848 033312 004037 037764      JSR   RO,GETR15     ;RESTORE R1-R5
7849 033316 000207              RTS   PC
7850 033320 004737 037136      6$: JSR   PC,SET.IE    ;SET "IE" WITHOUT A "TRE"
7851 033324 000770              BR     5$            ;EXIT THE ROUTINE
7852
7853 ;*COMMAND INITIATOR

```

```

7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864 033326 004737 037702
7865 033332 010237 032272
7866 033336 010203
7867 033340 013704 032360
7868 033344 010164 000010
7869 033350 062703 000004
7870 033354 062704 000002
7871 033360 012324
7872 033362 005037 001232
7873 033366 122737 000017 001102
7874 033374 001072
7875 033376 012337 001222
7876 033402 004037 037744
7877 033406 013701 001222
7878 033412 000301
7879 033414 042701 177400
7880 033420 006201
7881 033422 006201
7882 033424 006201
7883 033426 006201
7884 033430 016102 172340
7885 033434 005003
7886 033436 006302
7887 033440 006103
7888 033442 006302
7889 033444 006103
7890 033446 006302
7891 033450 006103
7892 033452 006302
7893 033454 006103
7894 033456 006302
7895 033460 006103
7896 033462 006302
7897 033464 006103
7898 033466 042737 160000 001222
7899 033474 050237 001222
7900 033500 013704 032360
7901 033504 005737 001234
7902 033510 001002
7903 033512 010364 000050
7904 033516 032703 000001 8$:
7905 033522 001403
7906 033524 052737 000400 001232
7907 033532 032703 000002 6$:
7908 033536 001403
7909 033540 052737 001000 001232

```

```

; *CALL
; * MOV #DRVNUM,R1 ;DRIVE NUMBER
; * MOV #DPB,R2 ;ADDRESS OF DPB
; * JSR PC,C1? ;C1?= C11,C13, OR C14
; * ;WHERE:
; * ;C11=DATA TRANSFER
; * ;C12=SEARCH REQUESTED BY DATA XFER
; * ;C14=NOT DATA TRANSFER
;
C11: JSR PC,POPQUE ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
MOV R2,TRNSWT ;PUT REQ. IN TRANSFER WAIT QUEUE
C12: MOV R2,R3 ;DPB ADDRESS TO R3
MOV RPADR,R4 ;RHCS1 ADDRESS
MOV R1,RHCS2(R4) ;SELECT DRIVE
ADD #4,R3 ;DESIRED WORD COUNT
ADD #2,R4 ;RHC ADDRESS
MOV (R3)+,(R4)+ ;LOAD WORD COUNT
CLR TEMP9
CMPB #17,$TSTNM
BNE 1$
MOV (R3)+,TEMP4
JSR R0,SAVR15
MOV TEMP4,R1
SWAB R1
BIC #177400,R1
ASR R1
ASR R1
ASR R1
ASR R1
MOV KIPARO(R1),R2
CLR R3
ASL R2
ROL R3
ASL R2
ROL R3
ASL R2
ROL R3
ASL R2
ROL R3
ASL R2
ROL R3
ASL R2
ROL R3
ASL R2
ROL R3
BIC #160000,TEMP4
BIS R2,TEMP4
MOV RPADR,R4
TST TEMP10
BNE 8$
MOV R3,RHBAE(R4)
BIT #1,R3
BEQ 6$
BIS #A16,TEMP9
BIT #2,R3
BEQ 7$
BIS #A17,TEMP9

```



```

7910 033546          75:
7911 033546 013764 001222 000004      MOV      TEMP4,RHBA(R4)
7912                                     :XYZ ABOVE R3 HAS VALUE FOR RHBAE IN 11/70
7913                                     :IN 11/45 LOWER 2 BITS MUST BE PUT IN A16 AND A17 POSITIONS IN 2(R2)
7914                                     :LOOK AT A FEW LINES DOWN AT LOCATION TAGED 55:
7915                                     :THAT 2(R2) MUST GET A16 A17 BY A BIS INSTRUCTION AND NOT A MOV INSTRUCTION
7916                                     :*****
7917                                     :*****
7918                                     :*****
7919                                     :*****
7920                                     :*****
7921                                     :*****
7922 033554 004037 037764      JSR      RO,GETR15
7923 033560 000401      BR      25
7924 033562 012324      15:     MOV      (R3)+,(R4)+      ;LOAD BUFFER ADDRESS
7925 033564 012346      25:     MOV      (R3)+,-(SP)   ;LOAD SECTOR AND TRACK
7926 033566 004037 036702      JSR      RO,WRT.RP      ;CALL THE LOAD(WRITE) ROUTINE
7927 033572 000006      RHDA    ;INDEX OF REGISTER TO LOAD
7928 033574 034402      CI7     ;ERROR RETURN ADDRESS
7929 033576 012346      MOV      (R3)+,-(SP)   ;LOAD CYLINDER ADDRESS
7930 033600 004037 036702      JSR      RO,WRT.RP
7931 033604 000034      RHCA
7932 033606 034402      CI7
7933 033610 122737 000017 001102      CMPB    #17,$TSTNM
7934 033616 001003      BNE     55
7935 033620 053762 001232 000002      BIS     TEMP9,2(R2)
7936 033626          55:
7937 033626 016246 000002      MOV      2(R2),-(SP)   ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
7938 033632 004037 036702      JSR      RO,WRT.RP
7939 033636 000000      RHCS1
7940 033640 034402      CI7
7941 033642 010137 032344      MOV      R1,DTUW      ;SET "DATA TRANSFER UNDERWAY"
7942 033646 000137 034336      JMP     CIS
7943 033652 013704 032360      CI3:   MOV      RPADR,R4      ;RHCS1 ADDRESS
7944 033656 010164 000010      MOV      R1,RHCS2(R4) ;SELECT DRIVE
7945 033662 016246 000012      MOV      12(R2),-(SP) ;DESIRED CYLINDER ADDRESS
7946 033666 004037 036702      JSR      RO,WRT.RP
7947 033672 000034      RHCA
7948 033674 034402      CI7
7949 033676 116203 000010      MOVB    10(R2),R3     ;PICKUP SECTOR ADDRESS
7950 033702 163703 032374      SUB     MXWINDW,R3    ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
7951 033706 002002      BGE     15
7952 033710 062703 000026      ADD     #22,R3
7953 033714 010346          15:     MOV      R3,-(SP)     ;COMBINE THE ADJUSTED SECTOR WITH
7954 033716 116266 000011 000001      MOVB    11(R2),1(SP)  ;THE DESIRED TRACK
7955 033724 004037 036702      JSR      RO,WRT.RP    ;LOAD DESIRED TRACK & SECTOR
7956 033730 000006      RHDA
7957 033732 034402      CI7
7958 033734 012746 000131      MOV      #131,-(SP)   ;START A SEARCH
7959 033740 004037 036702      JSR      RO,WRT.RP
7960 033744 000000      RHCS1
7961 033746 034402      CI7
7962 033750 156137 032346 032274      BISB    ATABIT(R1),SRCHWT ;SET "SEARCH WAIT" KEY
7963 033756 000567      BR     CIS
7964 033760 013704 032360      CI4:   MOV      RPADR,R4      ;RHCS1 ADDRESS
7965 033764 010164 000010      MOV      R1,RHCS2(R4) ;SELECT DRIVE

```


7966	033770	116203	000002		MOV B	2(R2),R3	: PICKUP THE REQUESTED COMMAND
7967	033774	122703	000131		CMP B	#131,R3	: IS IT A SEARCH COMMAND?
7968	034000	001007			BNE	1\$: BRANCH IF NO
7969	034002	016246	000010		MOV	10(R2),-(SP)	: LOAD DESIRED TRACK & SECTOR
7970	034006	004037	036702		JSR	RD,WRT.RP	
7971	034012	000006			RHDA		
7972	034014	034402			CI7		
7973	034016	000403			BR	2\$: GO LOAD CYLINDER
7974	034020	122703	000105	1\$:	CMP B	#105,R3	: IS IT A SEEK COMMAND
7975	034024	001007			BNE	3\$: BRANCH IF NO
7976	034026	016246	000012	2\$:	MOV	12(R2),-(SP)	: LOAD DESIRED CYLINDER
7977	034032	004037	036702		JSR	RD,WRT.RP	
7978	034036	000034			RHCA		
7979	034040	034402			CI7		
7980	034042	000546			BR	CI6	
7981	034044	122703	000115	3\$:	CMP B	#115,R3	: IS IT AN "OFFSET" COMMAND?
7982	034050	001013			BNE	4\$: BR IF NO
7983	034052	004037	036544		JSR	RD,RD.RP	: MERGE THE OFFSET VALUE INTO RHOF
7984	034056	000032			RHOF		: BUT DON'T CHANGE THE UPPER
7985	034060	034402			CI7		
7986	034062	116216	000001		MOV B	1(R2),(SP)	: BYTE WHEN LOADING THE
7987	034066	004037	036702		JSR	RD,WRT.RP	: REGISTER (RHOF)
7988	034072	000032			RHOF		
7989	034074	034402			CI7		
7990	034076	000530			BR	CI6	: GO START THE COMMAND
7991	034100	122703	000107	4\$:	CMP B	#107,R3	: IS IT A "RECALIBRATE" COMMAND?
7992	034104	001525			BEG	CI6	: BRANCH IF YES
7993	034106	122703	000117		CMP B	#117,R3	: IS IT A RETURN TO CENTER?
7994	034112	001522			BEG	CI6	: BRANCH IF YES
7995	034114	122703	000103		CMP B	#103,R3	: IS IT AN "UNLOAD" COMMAND?
7996	034120	001016			BNE	5\$: BRANCH IF NO
7997	034122	112761	000001	032232	MOV B	#1,DRVACT(R1)	: SET THE DRIVE ACTIVE INDICATOR
7998	034130	105061	032242		CLRB	DRVSTA(R1)	: PUT DRIVE STATUS TO OFFLINE
7999	034134	112761	000001	-032300	MOV B	#1,ULDFLG(R1)	: SET "UNLOAD IN PROGRESS" FLAG
8000	034142	010346			MOV	R3,-(SP)	: START THE "UNLOAD" COMMAND
8001	034144	004037	036702		JSR	RD,WRT.RP	
8002	034150	000000			RHCS1		
8003	034152	034402			CI7		
8004	034154	000207			RTS	PC	: RETURN TO USER
8005	034156	122703	000143	5\$:	CMP B	#143,R3	: IS IT A "SET FORMAT" COMMAND?
8006	034162	001014			BNE	6\$: BRANCH IF NO
8007	034164	004037	036544		JSR	RD,RD.RP	: READ THE OFFSET REGISTER
8008	034170	000032			RHOF		
8009	034172	034402			CI7		
8010	034174	116266	000001	000001	MOV B	1(R2),1(SP)	: COMBINE "FMT22", "ECI", AND "HCI"
8011	034202	004037	036702		JSR	RD,WRT.RP	: LOAD "FMT22", "ECI", AND/OR "HCI".
8012	034206	000032			RHOF		
8013	034210	034402			CI7		
8014	034212	000436			BR	12\$	
8015	034214	122703	000141	6\$:	CMP B	#141,R3	: IS IT A "GET REGISTER" COMMAND?
8016	034220	001023			BNE	10\$: BRANCH IF NO
8017	034222	016203	000006	7\$:	MOV	6(R2),R3	: POINTS TO 1ST ADDRESS OF WHERE
8018							: TO PUT THE REGISTER(S)
8019	034226	116237	000010	034244	MOV B	10(R2),9\$: INIT. THE INDEX FOR THE FIRST REG.
8020	034234	116205	000011		MOV B	11(R2),R5	: INDEX OF LAST REG. TO MOVE
8021	034240	004037	036544	8\$:	JSR	RD,RD.RP	: READ RP04 REGISTER

8022	034244	000000		95:	RHCS1		; INDEX OF REG. TO READ
8023	034246	034402			CI7		
8024	034250	012623			MOV	(SP)+, (R3)+	; GET THE CONTENTS OF RH11/RP04 REG.
8025	034252	023705	034244		CMP	95, R5	; LAST REG. BEEN READ?
8026	034256	001414			BEQ	125	; GET OUT IF YES
8027	034260	062737	000002	034244	ADD	#2, 95	; INCREASE THE INDEX BY 2
8028	034266	000764			SR	95	; LOOP--MORE TO READ
8029	034270	122703	000145	105:	CMPB	#145, R3	; IS IT A "SELECT DRIVE" COMMAND?
8030	034274	001405			BEQ	125	; BRANCH IF YES
8031	034276	010346		115:	MOV	R3, -(SP)	; LOAD THE COMMAND
8032	034300	004037	036702		JSR	RD, WRT.RP	
8033	034304	000000			RHCS1		
8034	034306	034402			CI7		
8035	034310	004737	037702	125:	JSR	PC, POPQUE	; REMOVE REQ. FROM QUEUE
8036	034314	052762	000200	000016	BIS	#BIT07, 16(R2)	; SET THE "DONE" BIT
8037	034322	005737	032320		TST	SAVEFG	; SAVE THE RH11/RP04 REGISTERS?
8038	034326	100002			BPL	135	; BRANCH IF NO
8039	034330	004737	037044		JSR	PC, SVRH11	; YES--GO SAVE THE REGISTERS
8040	034334	000207		135:	RTS	PC	; RETURN TO USER
8041	034336	006301		CI5:	ASL	R1	
8042	034340	012761	001750	032324	MOV	#1000., TIMER(R1)	; SET A ONE SECOND TIMER
8043	034346	006201			ASR	R1	
8044	034350	112761	000001	032232	MOVB	#1, DRVACT(R1)	; SET THE DRIVE ACTIVE
8045	034356	000207			RTS	PC	; RETURN TO THE USER
8046	034360	010346		CI6:	MOV	R3, -(SP)	; LOAD THE COMMAND
8047	034362	004037	036702		JSR	RD, WRT.RP	
8048	034366	000000			RHCS1		
8049	034370	034402			CI7		
8050	034372	000761			BR	CI5	
8051	034374	105761	032232	CI7A:	TSTB	DRVACT(R1)	; IS THE DRIVE ACTIVE?
8052	034400	001405			BEQ	CI7B	; BRANCH IF NO
8053	034402	012762	104000	000016	CI7:	MOV	#BIT15!BIT11, 16(R2)
8054	034410	004737	037044		JSR	PC, SVRH11	; SET "PARITY" ERROR INDICATOR
8055	034414	012746	000111	CI7B:	MOV	#111, -(SP)	; GO SAVE THE RH11/RP04 REGISTERS
8056	034420	004037	036702		JSR	RD, WRT.RP	; DO A "DRIVE CLEAR"
8057	034424	000000			RHCS1		
8058	034426	034466			CI8		
8059	034430	004737	037564		JSR	PC, EMPTYQ	; EMPTY THE QUEUE
8060	034434	105061	032300		CLRB	ULDFLG(R1)	; CLEAR THE UNLOAD IN QUEUE FLAG
8061	034440	105061	032232		CLRB	DRVACT(R1)	; DRIVE IS IDLE
8062	034444	020137	032344		CMP	R1, DTUW	; IF THIS DRIVE HAD AN I/O REQUEST
8063	034450	001005			BNE	15	; IN PROGRESS CLEAR ALL OF THE FLAGS
8064	034452	005037	032272		CLR	TRNSWT	
8065	034456	012737	177777	032344	MOV	#-1, DTUW	
8066	034464	000207		15:	RTS	PC	
8067	034466	004037	037744	CI8:	JSR	RD, SAVR15	; SAVE R1-R5
8068	034472	013704	032360		MOV	RPADR, R4	; PICKUP THE ADDRESS OF THE FIRST REGISTER
8069	034476	005001			CLR	R1	
8070	034500	005003			CLR	R3	
8071	034502	105761	032232	15:	TSTB	DRVACT(R1)	; DRIVE ACTIVE?
8072	034506	001421			BEQ	35	; BRANCH IF NO
8073	034510	013702	032272		MOV	TRNSWT, R2	; GET THE "TRANSFER WAIT" QUEUE
8074	034514	020137	032344		CMP	R1, DTUW	; DID THIS DRIVE HAVE AN I/O IN PROGRESS?
8075	034520	001402			BEQ	25	; BRANCH IF YES
8076	034522	004737	037660		JSR	PC, GETREQ	; GET THE DPB POINTER
8077	034526	004737	037044	25:	JSR	PC, SVRH11	; SAVE RH11/RP04 REGISTERS

E13

```

8078 034532 052762 102000 000016      BIS      #BIT15:BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
8079 034540 012763 177777 032324      MOV      #-1,TIMER(R3) ;STOP THE TIMER
8080 034546 105061 032232      CLR      DRVACT(R1) ;SET "DRIVE ACTIVE" TO IDLE
8081 034552 105061 032300      CLR      ULDFLG(R1) ;CLEAR UNLOAD FLAG
8082 034556 005201      INC      R1 ;MOVE TO THE NEXT DRIVE
8083 034560 062703 000002      ADD      #2,R3
8084 034564 042701 177770      BIC      #1<7,R1
8085 034570 001344      BNE      1$ ;BRANCH IF MORE DRIVES
8086 034572 012737 177777 032344      MOV      #-1,DTUW ;NO DATA TRANSFERS UNDERWAY
8087 034600 005037 032272      CLR      TRNSWT ;CLEAR THE "TRANSFER WAIT" QUEUE
8088 034604 004737 037502      JSR      PC,CLRQUE ;CLEAR ALL OF THE REQUEST QUEUES
8089 034610 012764 000040 000010      MOV      #BIT05,RHCS2(R4) ;DO A MASSBUS INIT.
8090 034616 004737 037136      JSR      PC,SET.IE ;SET "IE" WITHOUT "TRE"
8091 034622 004037 037764      JSR      RO,GETR15 ;RESTORE THE REGISTERS
8092 034626 000207      RTS      PC ;RETURN
8093
8094 ;*LOOK AHEAD ROUTINE
8095 ;*CALL
8096 ;*
8097 ;*      MOV      #DRVNUM,R1 ;DRIVE NUMBER
8098 ;*      MOV      #DPB,R2 ;POINT TO DPB
8099 ;*      JSR      RO,LA ;GO CHECK THE WINDOW
8100 ;*      RETURN1 ;ERROR RETURN
8101 ;*      RETURN2 ;START A SEARCH
8102 ;*      RETURN3 ;START A DATA TRANSFER
8103
8104 LA:      MOV      RPADR,R4 ;GET RHCS1'S ADDRESS
8105      MOV      R1,RHCS2(R4) ;SELECT DRIVE
8106      JSR      RO,RD.RP ;READ CURRENT CYLINDER
8107      RHCC
8108      4$ ;ERROR RETURN ADDRESS
8109      CMP      (SP)+,12(R2) ;IS CURRENT CYLINDER=DESIRED
8110 ;CYLINDER?
8111      BNE      3$ ;EXIT IF NO
8112      INCB      LACNT(R1) ;INCREMENT THE LOOK AHEAD COUNT
8113      CMP      LACNT(R1),MXLACT ;EXCEED MAX?
8114      BGT      2$ ;BRANCH IF YES
8115      MOV      10(R2),R3 ;GET DESIRED SECTOR ADDRESS AND
8116      SWAB      R3 ;MULT. BY 64--ALIGN WITH
8117      ASR      R3 ;LOOK AHEAD REGISTER
8118      ASR      R3
8119      MOV      #340,2#PS ;PRIORITY LEVEL "7"
8120      JSR      RO,RD.RP ;READ LOOK AHEAD REGISTER
8121      RHLA
8122      4$
8123      SUB      (SP)+,R3 ;CALCULATE THE DELTA
8124      BGE      1$
8125      ADD      #<22.*64.>,R3 ;MAKE THE DELTA POSITIVE
8126      CMP      MXDLTA,R3 ;CHECK THE DELTA TO SEE
8127      BLT      3$ ;IF IT IS WITHIN THE
8128      CMP      MNDLTA,R3 ;WINDOW---IF YES, ZERO
8129      BGE      3$ ;THE LOOK AHEAD COUNT
8130      CLR      LACNT(R1) ;AND TAKE THE I/O EXIT
8131      TST      (RO)+
8132      TST      (RO)+ ;ADJUST THE RETURN ADDRESS
8133      RTS      RO

```



```

8134 034760 004737 034402      4$:   JSR   PC,C17      ;PROCESS THE ERROR
8135 034764 000200                RTS   RO            ;TAKE ERROR RETURN
8136
8137                               ;*INTERRUPT SERVICE ROUTINE
8138
8139 034766 112737 000001 032276 ISR:   MOVB  #1,ACTDRV      ;SET "ACTIVE DRIVER" FLAG
8140 034774 004037 037744                JSR   RO,SAVR15     ;SAVE R1-R5
8141 035000 013704 032360                MOV  RPADR,R4      ;ADDRESS OF RHSCS1
8142 035004 013701 032344                MOV  DTUW,R1      ;GET "DATA TRANSFER UNDERWAY" INDICATOR
8143 035010 002403                BLT  1$           ;BRANCH IF NO DATA TRANSFER UNDERWAY
8144 035012 004737 035036                JSR   PC,TD        ;CALL TRANSFER DONE
8145 035016 000402                BR   2$           ;EXIT
8146 035020 004737 035164                1$:   JSR   PC,SC      ;CALL SPECIAL CONDITIONS
8147 035024 004037 037764                2$:   JSR   RO,GETR15 ;RESTORE R1-R5
8148 035030 105037 032276                CLR  ACTDRV       ;CLEAR "ACTIVE DRIVER" FLAG
8149 035034 000002                RTI                    ;RETURN
8150
8151                               ;*TRANSFER DONE ROUTINE
8152
8153 035036 105061 032232 TD:    CLR  DRVACT(R1) ;SET DRIVE ACTIVE INDICATOR TO IDLE
8154 035042 012737 177777 032344        MOV  #-1,DTUW     ;NO DATA TRANSFERS UNDERWAY
8155 035050 006301                ASL  R1
8156 035052 012761 177777 032324        MOV  #-1,TIMER(R1) ;CANCEL TIMEOUT
8157 035060 006201                ASR  R1
8158 035062 013702 032272                MOV  TRNSWT,R2    ;GET "DPB" ADDRESS FROM THE
8159 035066 005037 032272                CLR  TRNSWT       ;TRANSFER WAIT QUEUE--CLEAR QUEUE
8160 035072 052762 000200 000016        BIS  #BIT07,16(R2) ;SET DONE
8161 035100 010164 000010                MOV  R1,RHCS2(R4) ;SELECT THE DRIVE
8162 035104 004037 036544                JSR  RO,RD.RP     ;TRANSFER ERROR(TRE=1)?
8163 035110 000000                RHCS1
8164 035112 034402                C17
8165 035114 006126                ROL  (SP)+
8166 035116 100410                BMI  2$           ;BR IF YES
8167 035120 005737 032320                TST  SAVEFG       ;SAVE THE RH11/RP04 REGISTERS?
8168 035124 100002                BPL  1$           ;BRANCH IF NO
8169 035126 004737 037044                JSR  PC,SVRH11    ;YES--SAVE THE REGISTERS
8170 035132 004737 033162                1$:   JSR  PC,OPT     ;CALL OPTIMIZER
8171 035136 000456                BR   SC2          ;SPECIAL CONDITION (ENTRY #2)
8172 035140 052762 100100 000016        2$:   BIS  #BIT15:BIT06,16(R2) ;SET DATA ERROR FLAG
8173 035146 004737 037564                JSR  PC,EMPTYQ    ;EMPTY THE "DRIVES WAIT" QUEUE
8174 035152 004737 037044                JSR  PC,SVRH11    ;SAVE THE RH11/RP04 REGISTERS
8175 035156 012714 040111                MOV  #40111,(R4)  ;ISSUE A "DRIVE CLEAR"
8176 035162 000444                BR   SC2          ;SPECIAL CONDITION (ENTRY #2)
8177
8178                               ;*SPECIAL CONDITION ROUTINE
8179
8180 035164 005001 SC:    CLR  R1            ;START WITH DRIVE 0
8181 035166 012702                MOV  #1,R2
8182 035172 105761 032242                1$:   TST  DRVSTA(R1) ;NONEXISTENT?
8183 035176 002016                BGE  2$           ;NO--BRANCH
8184 035200 005201                INC  R1           ;YES--MOVE TO THE NEXT DRIVE
8185 035202 106302                ASLB R2          ;MORE DRIVES?
8186 035204 103372                BCC  1$          ;YES--BRANCH
8187 035206 013746 032322                MOV  SEEKFG,-(SP) ;SAVE THE "SEEK FLAG"
8188 035212 013746 032320                MOV  SAVEFG,-(SP) ;SAVE THE "SAVE FLAG"
8189 035216 004737 032376                JSR  PC,RPINIT   ;GO INIT. THE SUBSYSTEM

```

8190	035222	012637	032320		MOV	(SP)+,SAVEFG	;RESTORE THE "SAVE FLAG"
8191	035226	012637	032322		MOV	(SP)+,SEEKFG	;RESTORE THE "SEEK FLAG"
8192	035232	000405			BR	SC1	;TAKE ERROR EXIT
8193	035234	010164	000010	2\$:	MOV	R1,RHCS2(R4)	;SELECT DRIVE
8194	035240	116405	000001		MOVB	1(R4),R5	;IS "SC"=1?
8195	035244	100413			BMI	SC2	;BRANCH IF YES
8196	035246	012701	000010	SC1:	MOV	#8.,R1	;DO EIGHT DRIVES
8197	035252	005301		1\$:	DEC	R1	;NEXT DRIVE
8198	035254	002743			BLT	SC	;BRANCH IF OUT OF DRIVES
8199	035256	105761	032232		TSTB	DRVACT(R1)	;IS THIS DRIVE IDLE?
8200	035262	001373			BNE	1\$;BRANCH IF NO
8201	035264	104001			ERROR	1	;REPORT THE ERROR
8202	035266	004737	037136		JSR	PC,SET.IE	;GO SET INTERRUPT ENABLE
8203	035272	000207			RTS	PC	
8204	035274	012701	000003	SC2:	MOV	#3,R1	;READ RHAS UP TO THREE TIMES
8205	035300	116403	000016	1\$:	MOVB	RHAS(R4),R3	;READ "RHAS"
8206	035304	001011			BNE	2\$;BRANCH IF ANY ATA BITS = 1
8207	035306	005301			DEC	R1	;COUNT THIS READ
8208	035310	003373			BGT	1\$;LOOP IF MORE READS ALLOWED
8209	035312	004037	036544		JSR	RD,RD.RP	;READ CONTROL AND STATUS REGISTER
8210	035316	000000			RHCS1		
8211	035320	034466			CIB		
8212	035322	106126			ROLB	(SP)+	;IS "IE"=1?
8213	035324	100350			BPL	SC1	;NO--TAKE ERROR EXIT
8214	035326	000207			RTS	PC	;YES--RETURN
8215	035330	005046		2\$:	CLR	-(SP)	;PROCESS ALL DRIVES THAT HAVE
8216	035332	110316			MOVB	R3,(SP)	;AN "ATA"=1
8217	035334	012703	000001		MOV	#1,R3	
8218	035340	005001			CLR	R1	
8219	035342	030316		SC4:	BIT	R3,(SP)	;ATA=1?
8220	035344	001005			BNE	SC5	;YES--BRANCH
8221	035346	005201		SC3:	INC	R1	;MOVE TO THE NEXT DRIVE
8222	035350	106303			ASLB	R3	
8223	035352	001373			BNE	SC4	;BRANCH IF MORE TO CHECK?
8224	035354	005726			TST	(SP)+	;CLEAN OFF THE STACK
8225	035356	000207			RTS	PC	;RETURN TO USER
8226	035360	023701	032344	SC5:	CMP	DTUW,R1	;IS THIS DRIVE SETUP FOR I/O?
8227	035364	001002			BNE	1\$;NO---BRANCH
8228	035366	005726			TST	(SP)+	;YES---CLEAN OFF THE STACK (RHAS)
8229	035370	000622			BR	TD	;BRANCH TO "TRANSFER DONE"
8230	035372	105761	032242	1\$:	TSTB	DRVSTA(R1)	;CHECK THE DRIVE STATUS
8231	035376	003030			BGT	SC6	;BRANCH IF ONLINE
8232	035400	105761	032300		TSTB	ULDFLG(R1)	;UNLOAD IN PROGRESS?
8233	035404	003420			BLE	2\$;BRANCH IF NO
8234	035406	004737	037660		JSR	PC,GETREQ	;GET DPB POINTER
8235	035412	004737	037044		JSR	PC,SVRH11	;SAVE THE RH11/RP04 REGISTERS
8236	035416	004737	036074		JSR	PC,SC12	;SAVE RHDS1, RHER1, RHER2, AND RHER3
8237							;ALSO DO A DRIVE INIT (DRVINT)
8238	035422	105761	032242		TSTB	DRVSTA(R1)	;DID DRIVE COME ONLINE?
8239	035426	003411			BLE	3\$;NO---BRANCH
8240	035430	032737	040000	032222	BIT	#BIT:4,RPERRS	;WAS THERE AN ERROR?
8241	035436	001565			BEQ	SC11	;NO -- BRANCH
8242	035440	013705	032224		MOV	RPERRS+2,R5	;YES -- PICKUP RHER1 AND
8243	035444	000447			BR	SC6A	;GO PROCESS THE ERROR
8244	035446	004737	036074	2\$:	JSR	PC,SC12	;SAVE RHDS1, RHER1, RHER2, AND RHER3
8245							;ALSO DO A DRVINT

8246	035452	011605			3\$:	MOV	(SP),R5		;PICKUP (RHAS) BEFORE THE ERROR CALL
8247	035454	104005				ERROR	5		;REPORT THE ERROR
8248	035456	000733				BR	SC3		;GO CHECK FOR MORE ATA'S
8249	035460	006301			SC6:	ASL	R1		
8250	035462	012761	177777	032324		MOV	#-1,TIMER(R1)		;STOP THE TIMER
8251	035470	006201				ASR	R1		
8252	035472	004737	037660			JSR	PC,GETREQ		;GET THE DPB POINTER FROM THE QUEUE
8253	035476	010364	000016			MOV	R3,RHAS(R4)		;CLEAR ATTENTION
8254	035502	010164	000010			MOV	R1,RHCS2(R4)		;SELECT DRIVE
8255	035506	004037	036544			JSR	RO,RD.RP		;READ THE RP04'S STATUS REG.
8256	035512	000012				RHDS1			
8257	035514	035732				SCB			
8258	035516	011605				MOV	(SP),R5		;AND PUT IT IN R5
8259	035520	006126				ROL	(SP)+		;WAS THERE AN ERROR?
8260	035522	100115				BPL	SC9		;BR IF NO
8261	035524	105761	032232			TSTB	DRVACT(R1)		;CHECK THE DRIVE ACTIVE INDICATOR
8262	035530	001522				BEQ	SC10		;BRANCH IF IDLE
8263	035532	004037	036544			JSR	RO,RD.RP		;READ ERROR REGISTER #1
8264	035536	000014				RHER1			
8265	035540	035732				SCB			
8266	035542	012605				MOV	(SP)+,R5		;AND SAVE IT IN R5
8267	035544	004737	037044			JSR	PC,SVRH11		;SAVE RH11/RP04 REGISTERS
8268	035550	012746	000111			MOV	#111,-(SP)		;ISSUE A DRIVE CLEAR
8269	035554	004037	036702			JSR	RO,WRT.RP		
8270	035560	000000				RHCS1			
8271	035562	035732				SCB			
8272	035564	006105			SC6A:	ROL	R5		;WAS "UNSAFE" CONDITION =1?
8273	035566	100404				BMI	1\$;BRANCH IF YES
8274	035570	052762	100240	000016		BIS	#BIT15!BIT07!BIT05,16(R2)		;INFORM USER OF ERROR
8275	035576	000443				BR	SC7		
8276	035600	004037	036544		1\$:	JSR	RO,RD.RP		;READ DRIVE STATUS REG. #1
8277	035604	000012				RHDS1			
8278	035606	035732				SCB			
8279	035610	011605				MOV	(SP),R5		;SAVE RHDS1 IN R5
8280	035612	006126				ROL	(SP)+		; "ERR"=1?
8281	035614	100013				BPL	2\$;BR IF NO--UNSAFE CLEARED
8282	035616	112761	177777	032242		MOVB	#-1,DRVSTA(R1)		;DRIVE IS UNSAFE
8283	035624	004737	037044			JSR	PC,SVRH11		;SAVE RH11/RP04 REGISTERS
8284	035630	010364	000016			MOV	R3,RHAS(R4)		;CLEAR ATTENTION
8285	035634	052762	110000	000016		BIS	#BIT15!BIT12,16(R2)		;INFORM USER OF UNSAFE ERROR
8286	035642	000421				BR	SC7		
8287	035644	105705			2\$:	TSTB	R5		; "DRY" =1?
8288	035646	100414				BMI	3\$;BRANCH IF YES
8289	035650	112761	177777	032232		MOVB	#-1,DRVACT(R1)		;ACTIVE ERROR RECOVER
8290	035656	112761	000001	032242		MOVB	#1,DRVSTA(R1)		;ONLINE
8291	035664	006301				ASL	R1		
8292	035666	012761	072460	032324		MOV	#30000.,TIMER(R1)		;START 30 SECOND TIMER
8293	035674	006201				ASR	R1		
8294	035676	000623				BR	SC3		
8295	035700	052762	100220	000016	3\$:	BIS	#BIT15!BIT07!BIT04,16(R2)		;INFORM USER OF ERROR
8296	035706	105061	032232		SC7:	CLRB	DRVACT(R1)		;DRIVE IS IDLE
8297	035712	004737	037564			JSR	PC,EMPTYQ		;DUMP THE QUEUE
8298	035716	105761	032300			TSTB	ULDFLG(R1)		;UNLOAD IN PROGRESS OR QUEUE?
8299	035722	001611				BEQ	SC3		;BR IF NO
8300	035724	105061	032300			CLRB	ULDFLG(R1)		;CLEAR UNLOAD FLAG
8301	035730	000606				BR	SC3		

```

8302 035732 005726          SC8:  TST      (SP)+      ;REMOVE (RHAS) FROM THE STACK
8303 035734 105761 032232   TSTB     DRVACT(R1)  ;IS DRIVE IDLE?
8304 035740 001404          BEQ      1$          ;YES--BRANCH
8305 035742 004737 037660   JSR      PC,GETREQ  ;GET DPB POINTER
8306 035746 000137 034402   JMP      CI7        ;PROCESS THE PARITY ERROR
8307 035752 000137 034414   1$:     JMP      CI7B       ;PROCESS THE PARITY ERROR
8308 035756 105761 032232   SC9:    TSTB     DRVACT(R1)  ;TEST DRIVE ACTIVE
8309 035762 003013          BGT      SC11       ;BRANCH IF DRIVE IS ACTIVE
8310 035764 001404          BEQ      SC10       ;BRANCH IF DRIVE IS IDLE
8311 035766 052762 100210 000016  BIS     #BIT15:BIT07:BIT03,16(R2) ;INFORM USER OF ERROR RECOVER COMPLETION
8312 035774 000744          BR       SC7        ;
8313 035776 011605          SC10:   MOV      (SP),R5 ;PUT (RHAS) IN R5
8314 036000 004737 036074   JSR      PC,SC12    ;SAVE RHDS1, RHER1, RHER2, AND RHER3
8315 036004 104002          ERROR   2           ;REPORT THE ERROR
8316 036006 000137 035346   JMP      SC3        ;GO CHECK FOR MORE ATA'S
8317 036012 105761 032300   SC11:   TSTB     ULDFLG(R1) ;"UNLOAD IN PROGRESS"?
8318 036016 003402          BLE     1$          ;BRANCH IF NO
8319 036020 105061 032300   CLRB    ULDFLG(R1) ;CLEAR UNLOAD FLAG
8320 036024 105061 032232   1$:     CLRB    DRVACT(R1) ;SET DRIVE IDLE
8321 036030 136137 032346 032274  BITB    ATABIT(R1),SRCHWT ;DOING A SEARCH OPERATION FOR
8322                                     ;AN I/O COMMAND?
8323 036036 001012          BNE     2$          ;BRANCH IF YES
8324 036040 004737 037702   JSR      PC,POPQUE  ;REMOVE REQUEST FROM QUEUE
8325 036044 052762 000200 000016  BIS     #BIT07,16(R2) ;SET "DONE" BIT
8326 036052 005737 032320   TST     SAVEFG     ;SAVE THE RH11/RP04 REGISTERS?
8327 036056 100002          BPL     2$          ;BRANCH IF NO
8328 036060 004737 037044   JSR      PC,SVRH11 ;YES--SAVE ALL OF THE RH11/RP04 REG'S
8329 036064 004737 033162   2$:     JSR      PC,OPT  ;START A REQUEST
8330 036070 000137 035346   JMP      SC3        ;
8331 036074 010164 000010 032222  SC12:   MOV      R1,RHCS2(R4) ;SELECT DRIVE
8332 036100 016437 000012 032222  MOV     RHDS1(R4),RPERRS ;SAVE THE FOUR REGISTERS THAT
8333 036106 016437 000014 032224  MOV     RHER1(R4),RPERRS+2 ;WILL TELL US SOMETHING
8334 036114 016437 000040 032226  MOV     RHER2(R4),RPERRS+4
8335 036122 016437 000042 032230  MOV     RHER3(R4),RPERRS+6
8336 036130 004037 032540   JSR      RD,DRVINT ;INIT. THE STATE OF THE DRIVE
8337 036134 000401          BR      1$          ;TAKE ERROR EXIT
8338 036136 000207          RTS     PC          ;RETURN
8339 036140 005726          1$:     TST      (SP)+    ;POP PC OFF OF THE STACK
8340 036142 000673          BR      SC8        ;PROCESS THE PARITY ERROR
8341
8342 ;*RPO4 TIMER ROUTINE
8343 ;*CALL
8344 ;*   MOV     #TIME,-(SP) ;ELAPSED TIME IN MILLISECONDS ON THE STACK
8345 ;*   JSR     RD,RPTMR   ;CALL RPO4 TIME ROUTINE
8346
8347 036144 005737 032276   RPTMR:  TST      ACTDRV   ;CHECK "ACTDRV & ACTSTR"
8348 036150 001032          BNE     4$          ;IF NON ZERO EXIT
8349 036152 112737 000001 032277  MOVB    #1,ACTSTR   ;SET "ACTSTR"
8350 036160 004037 037744   JSR      RD,SAVR15  ;SAVE R1-R5
8351 036164 005001          CLR     R1         ;START WITH DRIVE 0
8352 036166 005003          CLR     R3
8353 036170 005763 032324   1$:     TST      TIMER(R3) ;IS THE TIMER RUNNING?
8354 036174 002407          BLT     2$          ;BRANCH IF NO
8355 036176 166663 000016 032324  SUB     16(SP),TIMER(R3) ;COUNT THE INTERVAL
8356 036204 003003          BGT     2$          ;BR IF NO SOFTWARE TIMEOUT
8357 036206 004037 036242   JSR      RD,STO    ;CALL SOFTWARE TIMEOUT ROUTINE

```



```

8358 036212 000405          BR      3$          ;GO TO THE EXIT
8359 036214 005201          2$:    INC      R1          ;MOVE TO NEXT DRIVE
8360 036216 005723          TST     (R3)+
8361 036220 022701 000010    CMP     #8.,R1        ;OUT OF DRIVES?
8362 036224 003361          BGT     1$          ;BRANCH IF NO
8363 036226 004037 037764    3$:    JSR     RD,GETR15 ;RESTORE R1-R5
8364 036232 105037 032277    CLRB   ACTSTR        ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
8365 036236 012616          4$:    MOV     (SP)+,(SP) ;ADJUST THE STACK
8366 036240 000200          RTS     RD          ;RETURN
8367
8368          ;*SOFTWARE TIMEOUT ROUTINE
8369          ;*
8370          ;*CALL: STO
8371          ;*      MOV     #DRVNUM,R1 ;DRIVE NUMBER
8372          ;*      JSR     RD,STO    ;CALL--DRVACT MUST BE NONZERO
8373          ;*      RETURN
8374
8375 036242 013746 177776    STO:   MOV     2#PS, -(SP) ;SAVE THE PROCESSOR STATUS
8376 036246 013737 032364 177776    MOV     RPVEC+2, 2#PS ;SET THE "PS" TO RP04 BR LEVEL
8377 036254 004037 037744          JSR     RD,SAVR15    ;SAVE R1-R5
8378 036260 013704 032360          MOV     RPADR,R4    ;GET ADDRESS OF "RHCS1"
8379 036264 010164 000010          MOV     R1,RHCS2(R4);SELECT THE DRIVE
8380 036270 004037 036544          JSR     RD,RD.RP    ;READ "DRIVE STATUS REG"
8381 036274 000012          RHDS1
8382 036276 036536          STOS
8383 036300 105726          TSTB   (SP)+        ;IS "DRY"=1?
8384 036302 100471          BMI    ST02        ;BR IF YES
8385 036304 013702 032272    ST01: MOV     TRNSWT,R2   ;PICKUP TRANSFER WAIT QUEUE
8386 036310 020137 032344          CMP     R1,DTUW    ;TRANSFER UNDERWAY ON THIS DRIVE?
8387 036314 001402          BEQ    1$          ;BRANCH IF YES
8388 036316 004737 037660          JSR     PC,GETREQ   ;GET DPB ADDRESS
8389 036322 052762 101000 000016 1$:    BIS     #BIT15!BIT09,16(R2);SET THE ERROR FLAGS
8390 036330 004737 037044          JSR     PC,SVRH11   ;SAVE RH11/RP04 REGISTERS
8391 036334 012764 000040 000010          MOV     #BIT05,RHCS2(R4);"INIT" THE MASS BUS
8392 036342 105061 032232          CLRB   DRVACT(R1) ;DRIVE IS IDLE
8393 036346 105061 032300          CLRB   ULDFLG(R1) ;CLEAR THE UNLOAD FLAG
8394 036352 005001          CLR    R1          ;START WITH DRIVE 0
8395 036354 005003          CLR    R3
8396 036356 004037 032540    2$:    JSR     RD,DRVINT   ;INIT. THIS DRIVE
8397 036362 000465          BR     ST05        ;PARITY ERROR RETURN
8398 036364 105761 032232          TSTB   DRVACT(R1) ;DRIVE IDLE BEFORE THE INIT.?
8399 036370 001414          BEQ    4$          ;YES--BRANCH
8400 036372 013702 032272          MOV     TRNSWT,R2   ;GET TRANSFER WAIT QUEUE
8401 036376 023701 032344          CMP     DTUW,R1    ;WAS THERE I/O ON THIS DRIVE?
8402 036402 001402          BEQ    3$          ;YES--BRANCH
8403 036404 004737 037660          JSR     PC,GETREQ   ;GET THE DPB POINTER FROM QUEUE
8404 036410 052762 100400 000016 3$:    BIS     #BIT15!BIT08,16(R2);INFORM USER OF INIT.
8405 036416 105061 032232          CLRB   DRVACT(R1) ;SET DRIVE ACTIVE TO IDLE
8406 036422 105061 032300          CLRB   ULDFLG(R1) ;NO UNLOAD
8407 036426 012763 177777 032324 4$:    MOV     #-1,TIMER(R3);STOP THE TIMER
8408 036434 005723          TST     (R3)+      ;UPDATE THE INDEX
8409 036436 005201          INC     R1          ;INCREMENT THE DRIVE NUMBER
8410 036440 022701 000010          CMP     #8.,R1    ;LAST DRIVE BEEN CHECKED?
8411 036444 003344          BGT     2$          ;NO--LOOP
8412 036446 012737 177777 032344          MOV     #-1,DTUW   ;NO DATA TRANSFERS UNDERWAY
8413 036454 005037 032272          CLR     TRNSWT    ;CLEAR TRANSFER WAIT QUEUE
  
```

```

8414 036460 004737 037502          JSR   PC,CLRQUE      ;CLEAR ALL REQUEST QUEUES
8415 036464 000417                   BR    ST04           ;EXIT
8416 036466 016405 000016          ST02: MOV   RHAS(R4),R5 ;IS ATTENTION FOR THIS
8417 036472 136105 032346          BITB  ATABIT(R1),R5 ;DRIVE UP?
8418 036476 001011                   BNE   ST03           ;YES--BRANCH
8419 036500 020137 032344          CMP   R1,DTUW       ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
8420 036504 001277                   BNE   ST01           ;BR IF NO
8421 036506 004037 036544          JSR   RD,RD.RP      ;YES--CHECK "RDY"
8422 036512 000000                   RHCS1
8423 036514 036536                   ST05
8424 036516 105726                   TSTB  (SP)+
8425 036520 100271                   BPL   ST01           ;BR IF "RDY"=0
8426 036522 005720                   ST03: TST  (RD)+     ;ADJUST FOR THE PROPER RETURN
8427 036524 004037 037764          ST04: JSR   RD,GETR15 ;RESTORE R1-R5
8428 036530 012637 177776          MOV   (SP)+,2#PS    ;RESTORE PROCESSOR STATUS
8429 036534 000200                   RTS   RD             ;RETURN
8430 036536 004737 034466          ST05: JSR   PC,CIB   ;GO HANDLE THE PARITY ERROR
8431 036542 000770                   BR    ST04
8432
8433 ;*ROUTINE TO READ A RH11/RP04 REGISTER
8434 ;*CALL
8435 ;*
8436 ;*   JSR   RD,RD.RP   ;GO READ A REGISTER
8437 ;*   INDEX ;REG. INDEX FROM BASE
8438 ;*   ERRADR ;ERROR ADDRESS--PROCESS ERROR STARTING
8439 ;*           ;AT THIS ADDRESS
8440 ;*   RETURN ;CONTENTS OF REG. IS ON THE STACK
8441
8442 036544 013737 032356 036670 RD.RP: MOV   MCPEMX,RD.RP2 ;MAX. RETRYS ALLOWED
8443 036552 011646                   MOV   (SP),-(SP)   ;SAVE RD FOR RETURN
8444 036554 013737 032360 036570 RD.RP1: MOV   RPADR,RD.ADR ;FORM THE DESIRED ADDRESS
8445 036562 062037 036570          RD.ADR: ADD  (RD)+,RD.ADR ;USING THE BASE AND THE INDEX
8446 036566 013727                   RD.WRD: MOV   2(PC)+,(PC)+ ;READ THE DESIRED REGISTER OF THE RP04
8447 036570 000000                   RD.ADR: .WORD 0 ;ADDRESS IS FORMED HERE
8448 036572 000000                   RD.WRD: .WORD 0 ;REG. CONTENTS PUT HERE
8449 036574 013766 036572 000002 RD.WRD: MOV   RD.WRD,2(SP) ;RETURN IT TO THE USER
8450 036602 017746 173552          MOV   2RPADR,-(SP) ;READ RHCS1
8451 036606 032716 020000          BIT   #BIT13,(SP)  ;DID MCPE SET?
8452 036612 001002                   BNE   1$            ;BRANCH IF YES
8453 036614 022620                   CMP   (SP)+,(RD)+  ;ADJUST FOR RETURN
8454 036616 000200                   RTS   RD             ;RETURN IF NO
8455 036620
8456 036620 104003                   1$:  ERROR 3 ;REPORT THE ERROR
8457 036622 005737 032344          TST   DTUW         ;DATA TRANSFER UNDERWAY?
8458 036626 100405                   BMI   2$            ;NO--BRANCH
8459 036630 032716 040000          BIT   #BIT14,(SP) ;NO--"TRE"=1?
8460 036634 001402                   BEQ   2$            ;NO--BRANCH
8461 036636 005726                   TST  (SP)+         ;YES--CLEAN OFF THE STACK AND
8462 036640 000415                   BR   RD.RP3        ;TAKE THE FATAL ERROR EXIT
8463 036642 052716 040000          2$:  BIS   #BIT14,(SP) ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
8464 036646 000316                   SWAB (SP)          ;POSITION BEFORE WRITING
8465 036650 013737 032360 036664 RD.RP3: MOV   RPADR,3$ ;FORM ADDRESS OF HIGH BYTE
8466 036656 005237 036664          INC  3$
8467 036662 112637                   MOVB (SP)+,2(PC)+  ;WRITE THE HIGH BYTE OF RHCS1
8468 036664 000000                   3$:  .WORD 0 ;ADDRESS STORAGE
8469 036666 005327                   DEC  (PC)+         ;EXCEEDED MAX. RETRYS

```



```

8470 036670 000003          RD.RP2:  WORD 3
8471 036672 002335          BGE      RD.RP1          ;BRANCH IF NO
8472 036674 011000          RD.RP3:  MOV      (RD),RO ;FATAL ERROR EXIT
8473 036676 012616          MOV      (SP)+,(SP)
8474 036700 000200          RTS      RO
8475
8476          ;*ROUTINE TO WRITE A RH11/RP04 REGISTER
8477          ;*CALL
8478          ;*
8479          ;*      MOV      DATA,-(SP)          ;DATA TO BE LOADED ON THE STACK
8480          ;*      JSR      RO,WRT.RP          ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
8481          ;*      INDEX          ;INDEX OF THE REGISTER TO BE LOADED
8482          ;*      ERRADR          ;ADDRESS TO RETURN TO ON AN ERROR
8483          ;*      RETURN          ;ERROR FREE RETURN
8484
8485 036702 016637 000002 036770 WRT.RP:  MOV      2(SP),WRT.WD ;SAVE THE WORD TO WRITE
8486 036710 012616          MOV      (SP)+,(SP) ;ADJUST THE STACK
8487 036712 012037 036772          MOV      (RO)+,WRT.AD ;GET INDEX OF REGISTER TO BE WRITTEN
8488 036716 001020          BNE      2$          ;BRANCH IF NOT RHCSI
8489 036720 122737 000150 036770          CMPB    #150,WRT.WD ;IS THE COMMAND FOR DATA TRANSFERS?
8490 036726 002414          BLT      2$          ;YES--DON'T GET THE OLD A16&A17, & PSEL
8491 036730 004037 036544          JSR      RO,RD.RP ;NO---COMBINE A16&A17, & PSEL WITH
8492 036734 000000          RHCSI          ;THE COMMAND BEFORE SENDING IT TO
8493 036736 036754          1$          ;THE RH11/RP04
8494 036740 000316          SWAB    (SP)
8495 036742 042716 177770          BIC      #1C7,(SP)
8496 036746 112637 036771          MOVB   (SP)+,WRT.WD+1
8497 036752 000402          BR      2$
8498 036754 011000          1$:      MOV      (RO),RO ;TAKE THE ERROR EXIT
8499 036756 000200          RTS      RO
8500 036760 063737 032360 036772 2$:      ADD      RPADR,WRT.AD ;FORM THE ADDRESS OF THE DISK REG.
8501 036766 012737          MOV      (PC)+,2(PC)+ ;LOAD THE DESIRED REG.
8502 036770 000000          WRT.WD: .WORD 0 ;WORD TO WRITE GOES HERE
8503 036772 000000          WRT.AD: .WORD 0 ;ADDRESS IS FORMED HERE
8504 036774 004037 036544          JSR      RO,RD.RP ;CHECK FOR PARITY ERROR ON WRITE
8505 037000 000014          RHER1
8506 037002 037034          2$
8507 037004 032726 000010          BIT      #BIT03,(SP)+
8508 037010 001413          BEQ      3$          ;BRANCH IF "PAR=0"
8509 037012 016037 177776 037024          MOV      -2(RO),1$ ;PICKUP THE INDEX
8510 037020 004037 036544          JSR      RO,RD.RP ;READ THE REG.
8511 037024 000000          1$:      .WORD 0 ;REG. INDEX
8512 037026 037034          2$          ;RETURN TO THIS ADDRESS ON ERROR
8513 037030 104004          ERROR   4          ;REPORT THE ERROR
8514 037032 005726          TST     (SP)+ ;CLEAN OFF THE STACK
8515 037034 011000          2$:      MOV      (RO),RO ;TAKE THE "PARITY ON WRITE" ERROR EXIT
8516 037036 000200          RTS      RO
8517 037040 005720          3$:      TST     (RO)+ ;ADJUST FOR ERROR-FREE EXIT
8518 037042 000200          RTS      RO
8519
8520          ;*ROUTINE TO SAVE THE RH11/RP04 REGISTERS AS PER DPB+14
8521          ;*CALL
8522          ;*
8523          ;*      MOV      #DPBNUM,R2          ;DPB POINTER TO R2
8524          ;*      JSR      PC,SVRH1          ;SAVE THE DRIVES REG'S
8525

```

```

8526 037044 004037 037744 SVRH11: JSR RD,SAVR15 ;SAVE REG.'S R1-R5
8527 037050 013704 032360 MOV RPADR,R4
8528 037054 111264 000010 MOVB (R2),RHCS2(R4) ;SELECT DRIVE
8529 037060 016202 000014 MOV 14(R2),R2 ;GET THE ERROR TABLE POINTER
8530 037064 001421 BEQ 4$ ;EXIT IF 0
8531 037066 005003 CLR R3 ;COUNTER & POINTER
8532 037070 012705 000022 MOV #RHDB,R5 ;PROBLEM REGISTER
8533 037074 020305 1$: CMP R3,R5 ;REACHED RHDB?
8534 037076 001005 BNE 2$ ;BR IF NO
8535 037100 105764 000010 TSTB RHCS2(R4) ;CHECK "OR"
8536 037104 100402 BMI 2$ ;BRANCH IF RHDB CAN BE READ
8537 037106 005022 CLR (R2)+ ;ELSE SAVE IT AS 0'S
8538 037110 000403 BR 3$
8539 037112 010446 2$: MOV R4,-(SP) ;FORM RH11/RP04 ADDRESS THAT IS
8540 037114 060316 ADD R3,(SP) ;TO BE READ
8541 037116 013622 MOV @((SP)+,(R2)+ ;AND READ IT
8542 037120 005723 3$: TST (R3)+ ;MOVE TO NEXT REG INDEX
8543 037122 020327 000046 CMP R3,#RHEC2 ;DONE?
8544 037126 003762 BLE 1$ ;BRANCH IF NO
8545 037130 004037 037764 4$: JSR RD,GETR15 ;GET REGISTERS R1-R5
8546 037134 000207 RTS PC ;RETURN TO USER
8547
8548 ;*ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
8549 ;*CALL
8550 ;* MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
8551 ;* JSR PC,SET.IE ;SET "IE"
8552 ;*
8553 ;* RETURN
8554 037136 010446 SET.IE: MOV R4,-(SP) ;SAVE R4
8555 037140 013704 032360 MOV RPADR,R4 ;PICKUP ADDRESS OF RHCS1
8556 037144 010164 000010 MOV R1,RHCS2(R4) ;SELECT DRIVE
8557 037150 011446 MOV (R4),-(SP) ;READ RHCS1
8558 037152 052716 040000 BIS #BIT14,(SP) ;SET THE "TRE" BIT OF THE WORD READ
8559 037156 000316 SWAB (SP) ;ADJUST FOR DATO
8560 037160 112714 000100 MOVB #BIT06,(R4) ;SET "IE"
8561 037164 032764 010000 000010 BIT #BIT12,RHCS2(R4) ;IS "NED"=1?
8562 037172 001002 BNE 1$ ;YES--CLEAR "TRE"
8563 037174 005726 TST (SP)+ ;CLEAN OFF THE STACK
8564 037176 000402 BR 2$
8565 037200 112664 000001 1$: MOVB (SP)+,1(R4) ;CLEAR "TRE"
8566 037204 012604 2$: MOV (SP)+,R4 ;RESTORE R4
8567 037206 000207 RTS PC ;RETURN TO CALLER
8568
8569 ;*QUEUE COUNT
8570 037210 000 QCNT: .BYTE 0 ;DRIVE 0
8571 037211 000 .BYTE 0 ;DRIVE 1
8572 037212 000 .BYTE 0 ;DRIVE 2
8573 037213 000 .BYTE 0 ;DRIVE 3
8574 037214 000 .BYTE 0 ;DRIVE 4
8575 037215 000 .BYTE 0 ;DRIVE 5
8576 037216 000 .BYTE 0 ;DRIVE 6
8577 037217 000 .BYTE 0 ;DRIVE 7
8578
8579 ;QUEUE INPUT POINTERS
8580
8581 037220 037302 QINPT: .WORD QDRVD ;DRIVE 0
  
```



```

8582 037222 037322 .WORD QDRV1 ;DRIVE 1
8583 037224 037342 .WORD QDRV2 ;DRIVE 2
8584 037226 037362 .WORD QDRV3 ;DRIVE 3
8585 037230 037402 .WORD QDRV4 ;DRIVE 4
8586 037232 037422 .WORD QDRV5 ;DRIVE 5
8587 037234 037442 .WORD QDRV6 ;DRIVE 6
8588 037236 037462 .WORD QDRV7 ;DRIVE 7
8589
8590 ;QUEUE OUTPUT POINTERS
8591
8592 037240 037302 QOUTPT: .WORD QDRV0 ;DRIVE 0
8593 037242 037322 .WORD QDRV1 ;DRIVE 1
8594 037244 037342 .WORD QDRV2 ;DRIVE 2
8595 037246 037362 .WORD QDRV3 ;DRIVE 3
8596 037250 037402 .WORD QDRV4 ;DRIVE 4
8597 037252 037422 .WORD QDRV5 ;DRIVE 5
8598 037254 037442 .WORD QDRV6 ;DRIVE 6
8599 037256 037462 .WORD QDRV7 ;DRIVE 7
8600
8601 037260 037302 QSTART: .WORD QDRV0 ;DRIVE 0 START ADDRESS
8602 037262 037322 QSTOP: .WORD QDRV1 ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
8603 037264 037342 .WORD QDRV2 ;STOP DRIVE 1--START DRIVE 2
8604 037266 037362 .WORD QDRV3 ;STOP DRIVE 2--START DRIVE 3
8605 037270 037402 .WORD QDRV4 ;STOP DRIVE 3--START DRIVE 4
8606 037272 037422 .WORD QDRV5 ;STOP DRIVE 4--START DRIVE 5
8607 037274 037442 .WORD QDRV6 ;STOP DRIVE 5--START DRIVE 6
8608 037276 037462 .WORD QDRV7 ;STOP DRIVE 6--START DRIVE 7
8609 037300 037502 .WORD QTERM ;STOP DRIVE 7
8610
8611 ;DRIVE REQUEST QUEUES
8612
8613 037302 000010 QDRV0: .BLKW 10
8614 037322 000010 QDRV1: .BLKW 10
8615 037342 000010 QDRV2: .BLKW 10
8616 037362 000010 QDRV3: .BLKW 10
8617 037402 000010 QDRV4: .BLKW 10
8618 037422 000010 QDRV5: .BLKW 10
8619 037442 000010 QDRV6: .BLKW 10
8620 037462 000010 QDRV7: .BLKW 10
8621 037502 000010 QTERM=.
8622
8623 ;*ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
8624
8625 ;*CALL
8626 ;*
8627 JSR PC,CLRQUE
8628 037502 004037 037744 CLRQUE: JSR R0,SAVR15 ;SAVE R1-R5
8629 037506 012702 037210 MOV #QCOUNT,R2 ;ZERO THE QUEUE COUNTS
8630 037512 005022 CLR (R2)+ ;DRIVES 0 & 1
8631 037514 005022 CLR (R2)+ ;DRIVES 2 & 3
8632 037516 005022 CLR (R2)+ ;DRIVES 4 & 5
8633 037520 005022 CLR (R2)+ ;DRIVES 6 & 7
8634 037522 012703 000010 MOV #8,R3 ;MOVE THE STARTING
8635 037526 012701 037260 MOV #QSTART,R1 ;ADDRESS OF THE QUEUE INTO
8636 037532 012122 1$: MOV (R1)+,(R2)+ ;THE QUEUE INPUT POINTER
8637 037534 005303 DEC R3
  
```



```

8638 037536 001375      BNE      1$
8639 037540 012703 000010      MOV      #8,R3      ;MOVE THE STARTING ADDRESS
8640 037544 012701 037260      MOV      #QSTART,R1 ;OF THE QUEUE INTO THE
8641 037550 012122      2$:      MOV      (R1)+,(R2)+ ;QUEUE OUTPUT POINTER
8642 037552 005303      DEC      R3
8643 037554 001375      BNE      2$
8644 037556 004037 037764      JSR      RD,GETR15  ;RESTORE R1-R5
8645 037562 000207      RTS      PC
8646
8647      ;*EMPTY THE QUEUE SPECIFIED BY R1
8648      ;
8649      ;*CALL
8650      ;*      MOV      DRVNUM,R1      ;DRIVE NUMBER TO R1
8651      ;*      JSR      PC,EMPTYQ
8652
8653 037564 105061 037210      EMPTYQ: CLRB   QCNT(R1)      ;CLEAR NUMBER OF ITEMS IN QUEUE
8654 037570 006301      ASL      R1
8655 037572 016161 037220 037240      MOV      QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
8656 037600 006201      ASR      R1
8657 037602 000207      RTS      PC
8658
8659      ;*ROUTINE TO PUT A REQUEST IN QUEUE
8660      ;
8661      ;*CALL
8662      ;*      MOV      #DRVNUM,R1      ;DRIVE NUMBER
8663      ;*      MOV      #DPB,R2      ;ADDRESS OF PARAMETER BLOCK
8664      ;*      JSR      RD,DRVQUE      ;GO PUT REQUEST IN QUEUE
8665      ;*      RETURN1      ;RETURN HERE IF QUEUE IS FULL
8666      ;*      RETURN2      ;RETURN HERE IF REQUEST IS IN QUEUE
8667
8668 037604 122761 000010 037210      DRVQUE: CMPB   #10,QCNT(R1)      ;IS QUEUE FULL?
8669 037612 001421      BEQ      2$      ;BR IF YES-TAKE RETURN1
8670 037614 105261 037210      INCB   QCNT(R1)      ;INCREMENT QUEUE COUNT
8671 037620 006301      ASL      R1
8672 037622 010271 037220      MOV      R2,QINPT(R1) ;PUT THIS REQUEST IN QUEUE
8673 037626 062761 000002 037220      ADD     #2,QINPT(R1) ;UPDATE THE QUEUE POINTER
8674 037634 026161 037220 037262      CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
8675 037642 001003      BNE     1$      ;BRANCH IF NO
8676 037644 016161 037260 037220      MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
8677 037652 006201      1$:      ASR     R1
8678 037654 005720      TST     (R0)+      ;TAKE RETURN 2
8679 037656 000200      2$:      RTS     R0      ;RETURN TO USER
8680
8681      ;*ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
8682      ;
8683      ;*CALL
8684      ;*      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
8685      ;*      JSR      PC,GETREQ      ;GO GET THE REQUEST
8686      ;*      RETURN      ;R2="DPB" ADDRESS OF THE REQUEST
8687      ;*      ;R2=0 IF NO REQUEST IN QUEUE
8688
8689 037660 005002      GETREQ: CLR     R2
8690 037662 105761 037210      TSTB   QCNT(R1)      ;IS THERE ANY REQUEST IN QUEUE?
8691 037666 001404      BEQ     2$      ;NO---BRANCH
8692 037670 006301      1$:      ASL     R1
8693 037672 017102 037240      MOV     QOUTPT(R1),R2 ;PICKUP "DPB" POINTER FOR THIS DRIVE

```



```

8694 037676 006201          ASR      R1
8695 037700 000207          RTS      PC          ;RETURN TO USER
8696
8697          ;*ROUTINE TO "POP" THE REQUEST FROM QUEUE
8698          ;*
8699          ;*CALL
8700          ;*      MOV      #DRVNUM,R1          ;DRIVE NUMBER TO R1
8701          ;*      JSR      PC,POPQUE          ;CALL TO REMOVE REQUEST
8702          ;*      RETURN          ;R2=ADDRESS OF DPB REMOVED
8703
8704 037702 105361 037210      POPQUE: DECB   QCNT(R1)          ;DECREMENT QUEUE COUNT
8705 037706 006301          ASL      R1
8706 037710 017102 037240      MOV      @QOUTPT(R1),R2          ;GET THE "DPB" POINTER
8707 037714 062761 000002 037240      ADD      #2,QOUTPT(R1)          ;UPDATE THE QUEUE POINTER
8708 037722 026161 037240 037262      CMP      QOUTPT(R1),QSTOP(R1)  ;TIME TO RESET THE POINTER?
8709 037730 001003          BNE      IS                      ;NO--BRANCH TO EXIT
8710 037732 016161 037260 037240      MOV      QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
8711 037740 006201          ASR      R1
8712 037742 000207          RTS      PC          ;RETURN TO USER
8713
8714          ;*ROUTINES TO SAVE R0-R5 AND R1-R5
8715          ;*
8716          ;*CALL: SAVROS
8717          ;*      JSR      R0,SAVROS
8718          ;*      RETURN          ;R0-R5 IS ON THE STACK
8719          ;*
8720          ;*CALL: SAVR15
8721          ;*      JSR      R0,SAVR15
8722          ;*      RETURN          ;R1-R5 IS ON THE STACK
8723          ;*
8724          ;*UPON RETURN FROM SAVROS AND SAVR15 THE STACK WILL LOOK LIKE:
8725          ;*
8726          ;*+12  R0
8727          ;*+10  R1
8728          ;*+06  R2
8729          ;*+04  R3
8730          ;*+02  R4
8731          ;*TOP  R5
8732
8733 037744          SAVROS:
8734 037744 010146          SAVR15: MOV      R1,-(SP)          ;SAVE R0 BY THE JSR INST.
8735 037746 010246          MOV      R2,-(SP)          ;SAVE R1
8736 037750 010346          MOV      R3,-(SP)          ;SAVE R2
8737 037752 010446          MOV      R4,-(SP)          ;SAVE R3
8738 037754 010546          MOV      R5,-(SP)          ;SAVE R4
8739 037756 016646 000012      MOV      12(SP),-(SP)       ;SAVE R5
8740 037762 000200          RTS      R0              ;GET R0
8741
8742          ;*ROUTINES TO RESTORE R0-R5 AND R1-R5
8743          ;*
8744          ;*CALL: GETROS
8745          ;*      JSR      R0,GETROS
8746          ;*      RETURN          ;R0-R5 HAVE BEEN RESTORED
8747          ;*
8748          ;*CALL: GETR15
8749          ;*      JSR      R0,GETR15

```

```

8750
8751
8752 037764 011666 000014
8753 037770 005726
8754 037772 012605
8755 037774 012604
8756 037776 012603
8757 040000 012602
8758 040002 012601
8759 040004 000200
8760
8761
8762
8763
8764 040006 000122
8765 040010 041506 000
8766 040013 114 000103
8767 040016 041511 000
8768 040021 106 000124
8769 040024 052114 000
8770 040027 111 000124
8771 040032 051506 000
8772 040035 114 000123
8773 040040 040520 000124
8774 040044 000075
8775 040046 005015 047503 052116
8776 040054 047522 020114 053523
8777 040062 052111 044103 051505
8778 040070 000075
8779 040072 005015 044122 051503
8780 040100 036461 000
8781 040103 015 051012 053110
8782 040110 041505 000075
8783 040114 005015 044122 051120
8784 040122 047511 000075
8785
8786 040126 005015 051012 052117
8787 040134 052101 047511 040516
8788 040142 020114 050123 042505
8789 040150 020104 044524 042515
8790 040156 000123
8791 040160 005015 047412 042516
8792 040166 041440 046131 047111
8793 040174 042504 020122 042523
8794 040202 045505 052040 046511
8795 040210 051505 005015 025052
8796 040216 030040 020054 026061
8797 040224 031040 027054 027056
8798 040232 030464 020060 025052
8799 040240 000
8800 040241 015 025012 020052
8801 040246 030464 026060 032040
8802 040254 034460 020054 030064
8803 040262 026070 027056 030056
8804 040270 025040 000052
8805 040274 005015 040412 042526

```

```

;* RETURN ;R1-R5 HAVE BEEN RESTORED
GETR15: MOV (SP),14(SP) ;POSITION R0
GETR05: TST (SP)+ ;POP R0 OF THE JSR OFF OF THE STACK
MOV (SP)+,R5 ;RESTORE R5
MOV (SP)+,R4 ;RESTORE R4
MOV (SP)+,R3 ;RESTORE R3
MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1
RTS R0 ;RESTORE R0

```

.SBTTL ASCIZ MESSAGES

```

MSG.R: .ASCIZ /R/
MSG.FC: .ASCIZ /FC/
MSG.LC: .ASCIZ /LC/
MSG.IC: .ASCIZ /IC/
MSG.FT: .ASCIZ /FT/
MSG.LT: .ASCIZ /LT/
MSG.IT: .ASCIZ /IT/
MSG.FS: .ASCIZ /FS/
MSG.LS: .ASCIZ /LS/
MSG.PAT: .ASCIZ /PAT/
MSG.EQ: .ASCIZ /=/
MSG.CS: .ASCIZ <15><12>/CONTROL SWITCHES=/

MRHCS1: .ASCIZ <15><12>/RHCS1=/
MRHVEC: .ASCIZ <15><12>/RHVEC=/
MRHPRI: .ASCIZ <15><12>/RHPRIO=/

MSG7: .ASCIZ <15><12><12>/ROTATIONAL SPEED TIMES/

MSG10A: .ASCIZ <15><12><12>/ONE CYLINDER SEEK TIMES/<15><12>/** 0, 1, 2,...410 **/

MSG10B: .ASCIZ <15><12>/** 410, 409, 408,...0 **/

MSG11A: .ASCIZ <15><12><12>/AVERAGE SEEK TIMES/<15><12>/** 0 TO 136 **/

```


8806	040302	040522	042507	051440	
8807	040310	042505	020113	044524	
8808	040316	042515	006523	025012	
8809	040324	020052	020060	047524	
8810	040332	030440	033063	025040	
8811	040340	000052			
8812	040342	005015	025052	030440	MSG11B: .ASCIZ <15><12>/** 136 TO 0 **/
8813	040350	033063	052040	020117	
8814	040356	020060	025052	000	
8815	040363	015	005012	040515	MSG12A: .ASCIZ <15><12><12>/MAXIMUM SEEK TIMES/<15><12>/** 0 TO 410 **/
8816	040370	044530	052515	020115	
8817	040376	042523	045505	052040	
8818	040404	046511	051505	005015	
8819	040412	025052	030040	052040	
8820	040420	020117	030464	020060	
8821	040426	025052	000		
8822	040431	015	025012	020052	MSG12B: .ASCIZ <15><12>/** 410 TO 0 **/
8823	040436	030464	020060	047524	
8824	040444	030040	025040	000052	
8825					
8826	040452	005015	044515	036516	MSGMIN: .ASCIZ <15><12>/MIN=/ 000
8827	040460	000			
8828	040461	015	046412	054101	MSGMAX: .ASCIZ <15><12>/MAX=/ 015
8829	040466	000075			
8830	040470	005015	053101	036507	MSGAVG: .ASCIZ <15><12>/AVG=/ 000
8831	040476	000			
8832	040477	060	052440	000123	MSGOUS: .ASCIZ /0 US/ 060
8833	040504	041040	046105	053517	MBELOW: .ASCIZ / BELOW THE MINIMUM OF / 041040
8834	040512	052040	042510	046440	
8835	040520	047111	046511	046525	
8836	040526	047440	020106	000	
8837	040533	040	041101	053117	MABOVE: .ASCIZ / ABOVE THE MAXIMUM OF / 040
8838	040540	020105	044124	020105	
8839	040546	040515	044530	052515	
8840	040554	020115	043117	000040	
8841	040562	051440	042505	051513	MSGNUM: .ASCIZ / SEEKS TIMED/ 051440
8842	040570	052040	046511	042105	
8843	040576	000			
8844	040577	040	047516	020124	MSGNON: .ASCIZ / NOT TIMED/ 040
8845	040604	044524	042515	000104	
8846	040612	020040	000		MSG.SP: .ASCIZ / / ;TWO (2) SPACES 020040
8847					
8848					.SBTTL ERROR HEADER (EM) MESSAGES
8849					
8850	040615	111	046114	043505	EM1: .ASCIZ /ILLEGAL RH11(SC=0),INTERRUPT OCCURRED/ 111
8851	040622	046101	051040	030510	
8852	040630	024061	041523	030075	
8853	040636	026051	047111	042524	
8854	040644	051122	050125	020124	
8855	040652	041517	052503	051122	
8856	040660	042105	000		
8857	040663	111	046114	043505	EM2: .ASCIZ /ILLEGAL RPO4 INTERRUPT OCCURRED/ 111
8858	040670	046101	051040	030120	
8859	040676	020064	047111	042524	
8860	040704	051122	050125	020124	
8861	040712	041517	052503	051122	

8862	040720	042105	000		
8863	040723	115	051501	041123	EM3: .ASCIZ /MASSBUS PARITY ERROR(MCPE=1)/
8864	040730	051525	050040	051101	
8865	040736	052111	020131	051105	
8866	040744	047522	024122	041515	
8867	040752	042520	030475	000051	
8868	040760	040515	051523	052502	EM4: .ASCIZ /MASSBUS PARITY ERROR(PAR=1)/
8869	040766	020123	040520	044522	
8870	040774	054524	042440	051122	
8871	041002	051117	050050	051101	
8872	041010	030475	000051		
8873	041014	043117	046106	047111	EM5: .ASCIZ /OFFLINE, NON-EXISTENT, OR UNSAFE DRIVE INTERRUPTED/
8874	041022	026105	047040	047117	
8875	041030	042455	044530	052123	
8876	041036	047105	026124	047440	
8877	041044	020122	047125	040523	
8878	041052	042506	042040	044522	
8879	041060	042526	044440	052116	
8880	041066	051105	052522	052120	
8881	041074	042105	000		
8882	041077	122	030510	027461	EM10: .ASCIZ "RH11/RP04 FAILED TO RESPOND TO ADDRESSING"
8883	041104	050122	032060	043040	
8884	041112	044501	042514	020104	
8885	041120	047524	051040	051505	
8886	041126	047520	042116	052040	
8887	041134	020117	042101	051104	
8888	041142	051505	044523	043516	
8889	041150	000			
8890	041151	104	044522	042526	EM11: .ASCIZ /DRIVE SELECTED IS NOT ONLINE/
8891	041156	051440	046105	041505	
8892	041164	042524	020104	051511	
8893	041172	047040	052117	047440	
8894	041200	046116	047111	000105	
8895	041206	046511	051120	050117	EM12: .ASCIZ /IMPROPER HEADER DATA/
8896	041214	051105	044040	040505	
8897	041222	042504	020122	040504	
8898	041230	040524	000		
8899	041233	104	052101	020101	EM13: .ASCIZ /DATA COMPARE FAILURE/
8900	041240	047503	050115	051101	
8901	041246	020105	040506	046111	
8902	041254	051125	000105		
8903	041260	044504	045523	042440	EM17: .ASCIZ /DISK ERROR IN TIMING TEST/
8904	041266	051122	051117	044440	
8905	041274	020116	044524	044515	
8906	041302	043516	052040	051505	
8907	041310	000124			
8908	041312	046103	041517	020113	EM20: .ASCIZ /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
8909	041320	045450	030527	026461	
8910	041326	024520	047440	042526	
8911	041334	043122	047514	020127	
8912	041342	047111	052040	046511	
8913	041350	047111	020107	042524	
8914	041356	052123	000		
8915	041361	122	030510	027461	EM41: .ASCIZ "RH11/RP04 ERROR"
8916	041366	050122	032060	042440	
8917	041374	051122	051117	000	

8918	041401	106	052101	046101	EM46: .ASCIZ /FATAL WRITE CHECK ERROR/
8919	041406	053440	044522	042524	
8920	041414	041440	042510	045503	
8921	041422	042440	051122	051117	
8922	041430	000			
8923					
8924					.SBTTL STATUS/ERROR INDICATOR MESSAGES
8925					
8926	041431	117	043106	044514	MSG814: .ASCIZ /OFFLINE OR UNSAFE DRIVE REQUESTED/
8927	041436	042516	047440	020122	
8928	041444	047125	040523	042506	
8929	041452	042040	044522	042526	
8930	041460	051040	050505	042525	
8931	041466	052123	042105	000	
8932	041473	125	046116	040517	MSG813: .ASCIZ /UNLOADED DRIVE REQUESTED/
8933	041500	042504	020104	051104	
8934	041506	053111	020105	042522	
8935	041514	052521	051505	042524	
8936	041522	000104			
8937	041524	042520	051522	051511	MSG812: .ASCIZ /PERSISTENT UNSAFE/
8938	041532	042524	052116	052440	
8939	041540	051516	043101	000105	
8940	041546	040520	044522	054524	MSG811: .ASCIZ /PARITY ERROR OCCURRED/
8941	041554	042440	051122	051117	
8942	041562	047440	041503	051125	
8943	041570	042522	000104		
8944	041574	040506	040524	020114	MSG810: .ASCIZ /FATAL PARITY ERROR/
8945	041602	040520	044522	054524	
8946	041610	042440	051122	051117	
8947	041616	000			
8948	041617	123	043117	053524	MSG809: .ASCIZ /SOFTWARE TIMEOUT ON THIS DRIVE/
8949	041624	051101	020105	044524	
8950	041632	042515	052517	020124	
8951	041640	047117	052040	044510	
8952	041646	020123	051104	053111	
8953	041654	000105			
8954	041656	047523	052106	040527	MSG808: .ASCIZ /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
8955	041664	042522	052040	046511	
8956	041672	047505	052125	047440	
8957	041700	020116	047101	052117	
8958	041706	042510	020122	051104	
8959	041714	053111	000105		
8960	041720	051105	047522	020122	MSG806: .ASCIZ "ERROR OCCURRED DURING I/O OPERATION"
8961	041726	041517	052503	051122	
8962	041734	042105	042040	051125	
8963	041742	047111	020107	027511	
8964	041750	020117	050117	051105	
8965	041756	052101	047511	000116	
8966	041764	051105	047522	020122	MSG805: .ASCIZ "ERROR OCCURRED DURING NON-I/O OPERATION"
8967	041772	041517	052503	051122	
8968	042000	042105	042040	051125	
8969	042006	047111	020107	047516	
8970	042014	026516	027511	020117	
8971	042022	050117	051105	052101	
8972	042030	047511	000116		
8973	042034	047125	040523	042506	MSG804: .ASCIZ /UNSAFE OCCURRED/

9142	043610	001170	001116	001154	DT41:	.WORD	\$TMPD,\$ERRPC,\$REGO,CHKDRV
9143	043616	001272					
9144	043620	001170	001116	001154	DT42:	.WORD	\$TMPD,\$ERRPC,\$REGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
9145	043626	001272	003574	003604			
9146	043634	003606					
9147	043636	001170	001116	001154	DT43:	.WORD	\$TMPD,\$ERRPC,\$REGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
9148	043644	001272	003574	003604			
9149	043652	003606					
9150	043654	003610	003634	003636	DT43A:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42
9151	043662	001170	001116	001154	DT44:	.WORD	\$TMPD,\$ERRPC,\$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
9152	043670	001272	001306	001312			
9153	043676	001310					
9154	043700	003574	003604	003606	DT44A:	.WORD	RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
9155	043706	003632	003630	003602			
9156	043714	003610	003634	003636	DT44B:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42
9157	043722	001170	001116	001154	DT45:	.WORD	\$TMPD,\$ERRPC,\$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
9158	043730	001272	001306	001312			
9159	043736	001310					
9160	043740	003574	003604	003606	DT45A:	.WORD	RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
9161	043746	003632	003630	003602			
9162	043754	003610	003634	003636	DT45B:	.WORD	RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+2,RP.REG+4,RP.REG+22
9163	043762	003576	003600	003616			

.SBTTL DATA FORMAT (DF) TABLE

9164							
9165							
9166							
9167	043770	000001			DF1:	.WORD	1
9168	043772	002				.BYTE	2
9169	043773	000				.BYTE	0
9170							
9171	043774	000001			DF2:	.WORD	1
9172	043776	010				.BYTE	108
9173	043777	000				.BYTE	0
9174							
9175	044000	000001			DF3:	.WORD	1
9176	044002	004				.BYTE	4
9177	044003	000				.BYTE	0
9178							
9179	044004	000001			DF4:	.WORD	1
9180	044006	005				.BYTE	5
9181	044007	000				.BYTE	0
9182							
9183	044010	000001			DF10:	.WORD	1
9184	044012	002				.BYTE	2
9185	044013	000				.BYTE	0
9186							
9187	044014	000001			DF11:	.WORD	1
9188	044016	002				.BYTE	2
9189	044017	000				.BYTE	0
9190							
9191	044020	000002			DF12:	.WORD	2
9192	044022	007				.BYTE	7
9193	044023	160				.BYTE	160
9194	044024	042472				.WORD	DH12A
9195	044026	006				.BYTE	6
9196	044027	000				.BYTE	0
9197							

;NUMBER OF DATA HEADERS
 ;NUMBER OF WORDS IN DATA TABLE
 ;BOTH NUMBERS ARE OCTAL

;2 DH'S TO BE TYPED
 ;7 DATA WORDS FOLLOW THE 1ST DH
 ;WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL
 ;ADDRESS OF 2ND DH
 ;6 DATA WORDS FOLLOW THE 2ND DH
 ;ALL WORDS ARE OCTAL

9198	044030	000002	DF13:	.WORD	2	
9199	044032	007		.BYTE	7	
9200	044033	160		.BYTE	160	
9201	044034	042551		.WORD	DH13A	
9202	044036	005		.BYTE	5	
9203	044037	004		.BYTE	4	;WORD 3 IS DECIMAL
9204						
9205	044040	000000	DF14:	.WORD	0	
9206	044042	005		.BYTE	5	
9207	044043	004		.BYTE	4	;WORD 3 IS DECIMAL
9208						
9209	044044	000001	DF17:	.WORD	1	
9210	044046	010		.BYTE	↑DB	
9211	044047	000		.BYTE	0	
9212						
9213	044050	000002	DF21:	.WORD	2	
9214	044052	006		.BYTE	6	
9215	044053	060		.BYTE	60	
9216	044054	042773		.WORD	DH21A	
9217	044056	004		.BYTE	4	
9218	044057	014		.BYTE	14	
9219						
9220	044060	000000	DF22:	.WORD	0	
9221	044062	004		.BYTE	4	
9222	044063	014		.BYTE	14	
9223						
9224	044064	000001	DF41:	.WORD	1	
9225	044066	004		.BYTE	4	
9226	044067	000		.BYTE	0	
9227						
9228	044070	000001	DF42:	.WORD	1	
9229	044072	007		.BYTE	7	
9230	044073	000		.BYTE	0	
9231						
9232	044074	000002	DF43:	.WORD	2	
9233	044076	007		.BYTE	7	
9234	044077	000		.BYTE	0	
9235	044100	043155		.WORD	DH43A	
9236	044102	003		.BYTE	3	
9237	044103	000		.BYTE	0	
9238						
9239	044104	000003	DF44:	.WORD	3	
9240	044106	007		.BYTE	7	
9241	044107	160		.BYTE	160	
9242	044110	043203		.WORD	DH44A	
9243	044112	006		.BYTE	6	
9244	044113	000		.BYTE	0	
9245	044114	043260		.WORD	DH44B	
9246	044116	003		.BYTE	3	
9247	044117	000		.BYTE	0	
9248						
9249	044120	000003	DF45:	.WORD	3	
9250	044122	007		.BYTE	7	
9251	044123	160		.BYTE	160	
9252	044124	043203		.WORD	DH44A	
9253	044126	006		.BYTE	6	

M14

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DERPKC.P11 DATA FORMAT (DF) TABLE

MACY11 27(732) 14-OCT-76 10:36 PAGE 182

9254	044127	000	.BYTE	0
9255	044130	043306	.WORD	DH45A
9256	044132	006	.BYTE	6
9257	044133	000	.BYTE	0
9258				
9259			.EVEN	
9260	044134		BUFFER=.	
9261				
9262	000001		.END	

IC4	002216	2085#	3358	3378				
IC5	002234	2094#	3416	3417				
IC6	002252	2103#	3466					
INCCYL	025356	4105	4390	6333#				
INCSK	006254	3266#	3270					
INCTRK	025326	4143	4428	6314#				
IOTVEC=	000020	1724#	2935*	2936*				
ISR	034766	3580	3671	3764	3856	6124	7700	8139#
IT	001506	1947#						
ITEM41	004064	2821#	4581					
IT15	002344	2146#	6316					
KIPAR0=	172340	1758#	4313*	4736	5591	5619	7884	
KIPAR1=	172342	1759#	4314*					
KIPAR2=	172344	1760#	4315*					
KIPAR3=	172346	1761#	4316*					
KIPAR4=	172350	1762#	4317*					
KIPAR5=	172352	1763#	4318*	4349*	4350*	4355	4435*	
KIPAR6=	172354	1764#	4319*	4355*	4356*	4436*		
KIPAR7=	172356	1765#	4320*	5632				
KIPDR0=	172300	1747#	4324*					
KIPDR1=	172302	1748#	4325*					
KIPDR2=	172304	1749#	4326*					
KIPDR3=	172306	1750#	4327*					
KIPDR4=	172310	1751#	4328*					
KIPDR5=	172312	1752#	4329*					
KIPDR6=	172314	1753#	4330*					
KIPDR7=	172316	1754#	4331*					
LA	034630	7841	8104#					
LACNT	032310	7567#	8112*	8113	8130*			
LC	001476	1943#						
LCO	002122	2047#	3197					
LC1	002136	2055#	3231					
LC15	002334	2142#	6333	6336	6338			
LC16	002362	2155#	4210	6756				
LC2	002160	2066#	3269	3271				
LC3	002176	2075#	3308					
LC4	002214	2084#	3359	3361	3368			
LC5	002232	2093#	3411	3418				
LC6	002250	2102#	3450					
LDCMD	022660	3078	5760#					
LERADD	001236	1858#	2996	4828				
LKS	001406	1908#	5712	5744*				
LKV	001402	1907#	5742*	5743*				
LPB	001416	1912#	4526					
LPS	001414	1911#	4525	5674				
LPTAVL	001254	1865#	4523	5671*	5675*			
LP.AVL	022340	2958	5671#					
LS	001512	1949#						
LS1	002150	2060#	3227					
LS15	002350	2148#	4077	4362				
LT	001504	1946#						
LT1	002144	2058#	3229					
LT15	002342	2145#	6314	6317	6319			
MABOVE	040533	6276	8837#					
MBELOW	040504	6260	8833#					
MCPMX	032356	7626#	8442					

K15

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DERPKC.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 14-OCT-76 10:36 PAGE 194

RPVEC	032362	3532*	3580*	3627*	3671*	3709*	3764*	3801*	3856*	6124*	7631#	7679	7699	7701
RP.REG	003574	7781	8376	2537	2561	2577#	5920	9135	9144	9147	9150	9154	9156	9160
RP04	032766	9162	2513	2537	2561	2577#	5920	9135	9144	9147	9150	9154	9156	9160
RP6	= 000300	5779	5811	5850	5893	5926	5945	5951	6131	7780#				
RSTRT1	005322	1659#												
RSTRT2	005334	3053	3055	3060#	3066									
RSTRT3	005342	3062#	3067											
RTC	= 000117	3063#												
RO	=%000000	1775#												
		1641#	2912*	2915	2916	2918*	2920	2921	2946*	2947*	2948	3010*	3012	3014
		3015*	3028*	3039	3049*	3082*	3084*	3088*	3102*	3117*	3200*	3201*	3234*	3235*
		3267*	3273*	3303*	3306*	3343*	3346*	3349*	3352*	3363*	3366*	3369*	3372*	3413*
		3415*	3455*	3457*	3459*	3461*	3463*	3465*	3515*	3567*	3583*	3587*	3617*	3658*
		3672*	3700*	3751*	3765*	3792*	3843*	3857*	3908*	3910*	3911*	3913*	3914*	3915*
		3919	3921*	3926*	3927*	3965*	3966	3967*	3969*	3970	3973*	3975	3976*	3978*
		3979	3981*	3982	3983	3990*	3994*	3997*	3998	4075*	4081*	4087	4102	4103
		4105*	4107*	4108	4127*	4131*	4139*	4143*	4202*	4205*	4216*	4222*	4223*	4224*
		4227*	4228*	4233*	4234*	4237*	4238*	4243*	4360*	4366*	4372	4387	4388	4390*
		4392*	4393	4412*	4416*	4424*	4428*	4489*	4491	4493	4496	4566*	4567*	4568
		4574*	4575	4576*	4577*	4578*	4579*	4581*	4582*	4583	4645	4650	4653*	4654
		4658	4659	4702	4710*	4723	4726*	4727	4728*	4729*	4730*	4731*	4732*	4733*
		4736	4762*	4876	4877*	4878	4881*	4989	4999*	5003	5019	5020	5033*	5248
		5273*	5284	5285*	5286	5287*	5288*	5289*	5327*	5331	5341*	5343*	5345*	5384*
		5385	5403*	5407*	5442	5443*	5444	5446	5448*	5449	5452*	5483	5490*	5493*
		5495*	5500*	5507*	5508	5511*	5518*	5519	5523*	5530	5534*	5550	5554*	5558*
		5563*	5567*	5573*	5575	5580*	5601	5607*	5647	5655*	5748*	5779*	5797*	5798*
		5811*	5829*	5833*	5837*	5839*	5850*	5868*	5872*	5876*	5878*	5893*	5926*	5945*
		5951*	5974*	5977*	5979*	5998	6000*	6001	6003	6006	6009	6012	6015	6018
		6021	6030*	6044	6054	6059*	6093	6106	6108*	6110*	6131*	6149	6153*	6170*
		6172*	6173	6175*	6237	6303*	6320	6321*	6339	6340*	6351*	6352*	6355	6358*
		6359	6361*	6372*	6386*	6387*	6398*	6448*	6450*	6451*	6491*	6497	6516*	6528*
		6535	6572*	6583	6596*	6599*	6629*	6632*	6657*	6692*	6693*	6714*	6769*	6789*
		6811*	6813	6814*	6818*	6820	6821*	6825*	6827	6832*	6836*	6838*	6840*	6845*
		6847	6848	6849	6884*	6892*	6897	6898	6910	6911	6917*	6926*	6934*	6944*
		6956*	6995*	6999*	7013*	7017*	7071*	7078*	7094*	7095	7103	7124*	7144*	7159*
		7174*	7182*	7183*	7190	7198*	7206*	7213*	7214	7226*	7230*	7233*	7234	7249*
		7254*	7255*	7257	7264	7268	7272*	7291	7292*	7311	7312*	7337*	7339	7340
		7341	7342	7343	7345*	7346*	7368*	7381*	7390*	7405*	7406	7411*	7412*	7428*
		7438*	7443*	7458*	7460*	7461*	7680*	7694*	7702*	7721*	7732*	7742*	7746*	7749*
		7755*	7766	7767*	7768*	7783*	7784	7792*	7798*	7808	7809*	7812*	7822*	7841*
		7848*	7876*	7922*	7926*	7930*	7938*	7946*	7955*	7959*	7970*	7977*	7983*	7987*
		8001*	8007*	8011*	8021*	8032*	8047*	8056*	8067*	8091*	8106*	8120*	8131	8132
		8133*	8135*	8140*	8147*	8162*	8209*	8255*	8263*	8269*	8276*	8336*	8350*	8357*
		8363*	8366*	8377*	8380*	8396*	8421*	8426	8427*	8429*	8445	8453	8454*	8472*
		8474*	8487	8491*	8498*	8499*	8504*	8509	8510*	8515*	8516*	8517	8518*	8526*
		8545*	8628*	8644*	8678	8679*	8740*	8759*						
RI	=%000001	1642#	2913*	2914	2915*	2919*	2920*	3011*	3016*	3029*	3036*	3043	3103*	3105*
		3107*	3109*	3111*	3304*	3305	3307*	3308	3340*	3342	3358*	3359	3361*	3362
		3378*	3379	3410*	3412	3416*	3418	3449*	3452	3454	3466*	3529*	3562*	3706*
		3746*	3798*	3838*	3916*	3920	3928*	3929	3983*	3985*	4077*	4078*	4080*	4083*
		4090*	4091*	4093*	4100*	4111	4114	4220*	4221	4362*	4363*	4365*	4368*	4375*
		4376*	4378*	4385*	4396	4399	4569	4575*	4588*	4590*	4591*	4592*	4600*	4607
		4650*	4662	4690	4990	5003*	5004	5008	5032*	5249	5272*	5323*	5327	5328*
		5340*	5342*	5344*	5386*	5392*	5398*	5484	5489*	5494*	5512*	5515	5519*	5528*
		5529	5533*	5551	5555*	5559*	5564*	5565*	5568*	5570*	5574*	5576	5579*	5602

5609*	5640*	5642	5643	5645*	5650	5654*	5699	5700*	5701	5702*	5703	5704*
5711*	5717*	5718*	5719*	5999	6002*	6023*	6024*	6025*	6026*	6027*	6028	6029*
6092	6093*	6095	6097	6102	6103	6104	6109*	6122*	6171*	6173	6333	6335*
6336	6338*	6353*	6355*	6373*	6376	6377	6378	6379	6380	6381	6382	6383
6400*	6406	6408	6410	6412	6414	6416	6418	6420	6422	6424	6426	6428
6430	6432	6434	6436	6441*	6442	6492*	6495*	6498*	6499*	6500*	6501*	6502*
6503*	6504*	6505*	6506*	6507*	6508*	6509*	6510*	6511*	6512*	6513*	6529*	6532*
6537*	6539*	6541*	6543*	6545*	6547*	6549*	6551*	6553*	6555*	6557*	6559*	6561*
6563*	6565*	6567*	6574	6586*	6587	6589	6624*	6627*	6647*	6651*	6653*	6654
6688	6709*	6710*	6726*	6729	6731	6734*	6749	6766	6841*	6844	6846*	6847*
6848*	6849*	6851	6883*	6901	6904	6906	6908	6933*	6941*	6951	6965*	6966
6968	6970	6975	6979	6983	6987	6991	6998*	7001*	7010	7012*	7016*	7019*
7035*	7038	7042	7046	7048*	7049	7054*	7055	7057	7063*	7064	7123*	7131
7134	7137	7139	7158*	7166	7168	7170	7197*	7217	7219	7221	7232*	7236
7256*	7258	7264*	7267	7285	7287	7289	7305	7307	7309	7329	7331	7333
7335	7344*	7398	7401	7430*	7451	7453*	7456	7682*	7684*	7685	7688*	7689
7691*	7696*	7697*	7705*	7722	7723	7725*	7754	7763*	7765*	7786*	7787	7789
7794	7802*	7803	7824	7828	7868	7877*	7878*	7879*	7880*	7881*	7882*	7883*
7884	7941	7944	7962	7965	7997*	7998*	7999*	8041*	8042*	8043*	8044*	8051
8060*	8061*	8062	8069*	8071	8074	8080*	8081*	8082*	8084*	8105	8112*	8113
8130*	8142*	8153*	8155*	8156*	8157*	8161	8180*	8182	8184*	8193	8196*	8197*
8199	8204*	8207*	8218*	8221*	8226	8230	8232	8238	8249*	8250*	8251*	8254
8261	8282*	8289*	8290*	8291*	8292*	8293*	8296*	8298	8300*	8303	8308	8317
8319*	8320*	8321	8331	8351*	8359*	8361	8379	8386	8392*	8393*	8394*	8398
8401	8405*	8406*	8409*	8410	8417	8419	8556	8635*	8636	8640*	8641	8653*
8654*	8655*	8656*	8668	8670*	8671*	8672*	8673*	8674	8676*	8677*	8690	8692*
8693	8694*	8704*	8705*	8706	8707*	8708	8710*	8711*	8734	8758*		
1643*	2914*	2916	3069*	3070	3074	3075	3076	3077	3093	3104*	3106*	3108*
3110*	3112*	3113*	3114	3335*	3338*	3344	3350	3364	3370	3411*	3414	3417*
3420	3450*	3451*	3452	3541*	3555*	3570*	3637*	3661*	3719*	3737*	3754*	3811*
3829*	3846*	4110*	4112	4116	4395*	4397	4401	4570	4593*	4599	4654*	4675
4696*	4724	4736*	4738*	4740*	4742*	4744*	4746*	4748*	4752	4761*	4991	5002*
5006*	5009	5016*	5017*	5018	5023*	5031*	5250	5271*	5329*	5330	5331*	5346*
5347*	5383*	5386	5387*	5393*	5394*	5399*	5400*	5485	5496*	5504*	5509	5532*
5552	5557*	5560*	5566*	5569*	5571*	5572*	5573	5578*	5603	5619*	5621*	5622*
5625*	5626*	5631*	5632	5634*	5639*	5641*	5646*	5651	5653*	6193*	6196*	6197
6199*	6202*	6203	6204	6206*	6209*	6210	6211*	6212*	6213*	6237*	6240	6241
6242	6243*	6277	6314	6316*	6317	6319*	6354*	6356*	6374*	6376*	6377*	6378*
6379*	6380*	6381*	6382*	6383*	6384	6403*	6438*	6496*	6514*	6533*	6569*	6628*
6630*	6646*	6650*	6651	6652*	6655	6656	6661*	6664*	6665	6667	6669	6705
6706*	6711*	6713*	6727*	6736	6738	6741*	6751	6767	6836	6840	6899	6912*
6913*	6914*	6915*	6916*	6943*	6950	6955	6997	7005*	7006	7008*	7009	7015
7023*	7024	7026*	7027	7030	7031	7032	7096*	7101	7103*	7143*	7150	7179
7180*	7211	7224	7228	7239	7243	7257*	7260	7289*	7290*	7309*	7310*	7364
7365*	7380	7407*	7437	7441*	7683*	7685	7687*	7689	7784*	7785*	7786	7796*
7800	7807*	7826	7831*	7833	7865	7866	7884*	7886*	7888*	7890*	7892*	7894*
7896*	7899	7935*	7937	7945	7949	7954	7966	7969	7976	7986	8010	8017
8019	8020	8036*	8053*	8073*	8078*	8109	8115	8158*	8160*	8172*	8181*	8185*
8274*	8285*	8295*	8311*	8325*	8385*	8389*	8400*	8404*	8528	8529*	8537*	8541*
8629*	8630*	8631*	8632*	8633*	8636*	8641*	8672	8689*	8693*	8706*	8735	8757*
1644*	3524*	3527*	3620*	3703*	3795*	4119*	4120*	4121	4404*	4405*	4406	4571
4594*	4602*	4658*	4677	4682	4692*	4725	4737*	4739*	4741*	4743*	4745*	4747*
4749*	4753	4760*	4931	4940*	4946*	4947*	4950*	4955*	4956*	4957	4966*	4992
5000*	5001*	5015*	5018*	5027*	5028*	5030*	5119	5121*	5122	5125*	5126	5133*
5134	5136	5144	5148	5150*	5156	5158	5160*	5163*	5251	5270*	5326*	5339*
5346	5391*	5396*	5402*	5403	5486	5508*	5509*	5511	5531*	5553	5556*	5561*

R2 =%000002

R3 =%000003

M15

MAINDEC-11-DERPK-C "MECHANICAL AND READ/WRITE TEST"
DERPKC.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 14-OCT-76 10:36 PAGE 196

5577*	5604	5620*	5624*	5652*	6200	6207	6242*	6261	6375*	6384	6404*	6406
6408	6410	6412	6414	6416	6418	6420	6422	6424	6426	6428	6430	6432
6434	6436	6452	6461	6497*	6498	6499	6500	6501	6502	6503	6504	6505
6506	6507	6508	6509	6510	6511	6512	6513	6535*	6537	6539	6541	6543
6545	6547	6549	6551	6553	6555	6557	6559	6561	6563	6565	6567	6582
6586	6588	6590	6654*	6661	6664	6666	6668	6688	6728*	6743	6746*	6747*
6749*	6750*	6751*	6752*	6754*	6756	6759*	6761	6763*	6764*	6768	6787*	6812*
6819*	6826*	6877	6973*	7044*	7051	7060	7067	7082	7251	7258*	7262	7265
7363	7366*	7375*	7376	7377*	7378*	7379*	7380*	7403	7407	7409*	7426	7427*
7431*	7433*	7435*	7437*	7441	7442*	7450	7454	7455	7458	7459*	7699*	7700*
7701*	7723*	7724*	7726*	7736*	7866*	7869*	7871	7875	7885*	7887*	7889*	7891*
7893*	7895*	7897*	7903	7904	7907	7924	7925	7929	7949*	7950*	7952*	7953
7966*	7967	7974	7981	7991	7993	7995	8000	8005	8015	8017*	8024*	8029
8031	8046	8070*	8079*	8083*	8115*	8116*	8117*	8118*	8123*	8125*	8126	8128
8205*	8216	8217*	8219	8222*	8253	8284	8352*	8353	8355*	8360	8395*	8407*
8408	8531*	8533	8540	8542	8543	8634*	8637*	8639*	8642*	8736	8756*	
1645#	3518*	3521*	3535*	3538	3543*	3544*	3548*	3552	3557*	3558*	3572*	3573*
3577*	3578*	3629*	3630*	3634	3639*	3640*	3663*	3664*	3668*	3669*	3711*	3712*
3716	3721*	3722*	3729*	3730*	3734	3739*	3740*	3756*	3757*	3761*	3762*	3803*
3804*	3808	3813*	3814*	3821*	3822*	3826	3831*	3832*	3848*	3849*	3853*	3854*
4076*	4082*	4085*	4086	4113	4361*	4367*	4370*	4371	4398	4572	4629	4630*
4635	4641	4643*	4659*	4660*	4932	4934*	4935*	4936*	4937	4938*	4952	4954*
4962*	4965*	5252	5269*	5325*	5332*	5389*	5392	5398	5400	6125*	6126*	6151*
6152*	6244*	6247	6250	6254	6257	6266	6270	6273	6282	6283	6284	6294
6452*	6453*	6454	6455*	6574*	6577*	6579*	6581*	6669*	6672*	6673*	6675*	6676*
6788*	6792	6813*	6818	6820*	6825	6827*	6828*	6829*	6830*	6831*	6838	6880
6899*	6912	6974*	6977*	6981*	6985*	6989*	6993*	7009*	7027	7030*	7032	7036
7040	7044	7106*	7110	7112	7115	7119	7143	7150*	7155	7179*	7184*	7194
7211*	7212	7224*	7228*	7236*	7239*	7243*	7259*	7260	7262	7265	7269*	7270
7362	7367*	7388	7389	7397	7405	7410*	7692*	7693*	7722*	7728	7754*	7791*
7867*	7868*	7870*	7871*	7900*	7903*	7911*	7924*	7943*	7944*	7964*	7965*	8068*
8089*	8104*	8105*	8141*	8151*	8175*	8193*	8194	8205	8253*	8254*	8284*	8331*
8332	8333	8334	8335	8378*	8379*	8391*	8416	8527*	8528*	8535	8539	8554
8555*	8556*	8557	8560*	8561	8565*	8566*	8737	8755*				
1646#	3523*	3624*	3645	3650*	3725*	3743*	3817*	3835*	4086*	4089*	4092*	4095*
4118	4371*	4374*	4377*	4380*	4403	4573	4652*	4656*	4699*	4706	4933	4939*
4941*	4943*	4944*	4945*	4946	4964*	4993	4995*	4997*	5004*	5008*	5023	5029*
5253	5268*	5324*	5330*	5335*	5336	5390*	5394	5401	6194	6241*	6298	6300*
6454*	6456*	6458*	6459*	6460*	6481	6582*	6583*	6584*	6585*	6611	6997*	7002*
7003*	7004*	7005	7015*	7020*	7021*	7022*	7023	7080*	7081*	7082	7094	7246
7247	7251	7256	7735*	7736	7737	7739	7752*	7760*	7761*	8020*	8025	8194*
8242*	8246*	8258*	8266*	8272*	8279*	8287	8313*	8416*	8417	8532*	8533	8738
8754*												
R6 =%000006	1647#	1649	2930*	2931*	2932							
R7 =%000007	1648#	1650										
SAVEFG 032320	7582#	8037	8167	8188	8190*	8326						
SAVREG= 104416	3540	3554	3569	3636	3660	3718	3736	3753	3810	3828	3845	5307#
	5382	6169										5322
SAVROS 037744	4566	6351	6528	6926	7071	7182	7254	8733#				
SAVR15 037744	6372	6398	6491	7680	7721	7783	7822	7876	8067	8140	8350	9377
	8628	8734#										8526
SC 035164	8146	8180#	8198									
SCTRWC= 177400	1792#	3918	3963	4212								
SC1 035246	8192	8196#	8213									
SC10 035776	8262	8310	8313#									
SC11 036012	8241	8309	8317#									

R4 =%000004

R5 =%000005

SC12	036074	8236	8244	8314	8331*																	
SC2	035274	8171	8176	8195	8204*																	
SC3	035346	8221*	8248	8294	8299	8301	8316	8330														
SC4	035342	8219*	8223																			
SC5	035260	8220	8226*																			
SC6	035460	8231	8249*																			
SC6A	035564	8243	8272*																			
SC7	035706	8275	8286	8296*	8312																	
SC8	035732	8257	8265	8271	8278	8302*	8340															
SC9	035756	8260	8308*																			
SEARCH=	000131	1778*	3535	3548																		
SEC.DS	001310	1881*	5788*	5820*	5859*	5905*	5935*	5961*	6102*	6140*	6595*	9125	9128	9130								
		9151	9157																			
SEC.RD	001304	1879*	6101*	9128																		
SEEK =	000105	1770*	3630	3712	3730	3804	3822	5765	5766													
SEEKFG	032322	3018*	3021*	7590*	7683	7839	8187	8191*														
SELDV=	000145	1787*																				
SETBUF	026076	4123	4408	6491*																		
SETFOR=	000143	1786*	3080	3081	3083	3087	3194															
SET.IE	037136	7730	7850	8090	8202	8554*																
SP =	%000006	1649*	2934*	2975*	2976*	2977*	2978*	2979*	2980*	2981*	2982	3039*	3093*	3199*								
		3233*	3264*	3301*	3339*	3409*	3448*	3519*	3520*	3521	3528*	3567	3568	3622*								
		3658	3659	3705*	3751	3752	3797*	3843	3844	3900*	3958*	4215*	4323*	4324								
		4325	4326	4327	4328	4329	4330	4331	4341*	4342*	4343*	4344*	4345*	4346*								
		4347*	4352*	4354	4463*	4471*	4534	4546*	4549*	4629*	4635*	4641*	4643	4662*								
		4690*	4723*	4724*	4725*	4726	4754*	4756*	4757	4760	4761	4762	4763*	4764*								
		4782*	4789*	4811*	4822*	4828*	4834*	4836*	4876*	4877	4878*	4880	4881	4882*								
		4885	4887*	4889*	4895	4923*	4924	4925	4926*	4931*	4932*	4933*	4939	4964								
		4965	4966	4967*	4968*	4989*	4990*	4991*	4992*	4993*	4994*	4995	4998*	5011								
		5013*	5015	5025	5027	5029	5030	5031	5032	5033	5035*	5036*	5073*	5074*								
		5075	5083	5086	5099*	5100*	5101*	5106*	5119*	5120*	5125	5128	5132*	5139								
		5143*	5162	5163	5164*	5165*	5166*	5193*	5196	5198	5199	5221	5222	5226*								
		5248*	5249*	5250*	5251*	5252*	5253*	5254*	5255*	5256*	5257*	5264*	5265*	5266*								
		5267*	5268	5269	5270	5271	5272	5273	5284*	5285	5323	5336*	5362	5363*								
		5365*	5383	5385*	5442*	5443	5452	5453*	5481*	5482*	5483*	5484*	5485*	5486*								
		5487*	5488*	5489	5490	5492*	5496	5499*	5502*	5513*	5517*	5520	5521	5524*								
		5525*	5526	5529*	5530*	5531	5532	5533	5534	5535*	5550*	5551*	5552*	5553*								
		5577	5578	5579	5580	5601*	5602*	5603*	5604*	5605*	5606*	5607	5618*	5622								
		5623*	5647*	5648	5649	5652	5653	5654	5655	5677	5699*	5701*	5703*	5710								
		5716	5717	5718	5719	5747*	5789*	5791	5821*	5823	5860*	5862	5906*	5908								
		5936*	5938	5962*	5964	5998*	5999*	6000	6028*	6029	6030	6044*	6048*	6049*								
		6050*	6051*	6053*	6055*	6058	6072*	6074*	6079*	6080	6081	6092*	6109	6141*								
		6143	6250*	6257*	6261*	6266*	6273*	6277*	6282*	6283*	6284*	6286*	6288*	6294*								
		6399*	6401*	6402*	6404	6440*	6444	6449*	6453	6462	6468*	6534*	6571	6591								
		6596	6599	6600*	6609*	6644*	6645*	6658	6659	6660*	6677	6679*	6690	6705*								
		6713	6852	6880*	6883	6933	6965	7088*	7095*	7098*	7105*	7107*	7119*	7123								
		7155*	7158	7190*	7194*	7197	7240	7244	7362*	7363*	7364*	7376*	7379	7403								
		7408	7409	7410	7426*	7459	7678*	7703	7735	7741*	7745*	7752	7758*	7780*								
		7811	7823*	7847	7925*	7929*	7937*	7945*	7953*	7954*	7958*	7969*	7976*	7986*								
		8000*	8010*	8024	8031*	8046*	8055*	8109	8123	8165*	8187*	8188*	8190	8191								
		8212*	8215*	8216*	8219	8224	8228	8246	8258	8259*	8266	8268*	8279	8280*								
		8302	8313	8339	8355	8365*	8375*	8383	8424	8428	8443*	8449*	8450*	8451								
		8453	8459	8461	8463*	8464*	8467	8473*	8485	8486*	8494*	8495*	8496	8507								
		8514	8539*	8540*	8541	8554*	8557*	8558*	8559*	8563	8565	8566	8734*	8735*								
		8736*	8737*	8738*	8739*	8752*	8753	8754	8755	8756	8757	8758										
		7536*	7824*	7962*	8321																	

CKCHR	2903#	6884	6934	7124	7159	7198	7368	7381	7390	7443					
CKDIG	2903#	6942	7142												
CKNUM	2903#	6891	7173	7205											
COMMEN	1#	1731#	3123	3158	3472	3862									
COMND	2903#	3080	3083	3087	3194										
DO	2903#	3082	3084	3088	3200	3201	3234	3235	3267	3273	3303	3306	3343	3349	3363
	3369	3413	3415	3455	3457	3459	3461	3463	3465						
DODTA	2903#	3908	3910	3913	3921	3926	3967	3976	3990	3994	4127	4131	4139	4204	4223
	4227	4233	4237	4243	4412	4416	4424								
DRV. IN	2903#	3577	3668	3761	3853										
ENDCOM	1#	1731#	3151	3169	3485	3871									
ENDPAS	2903#	4484													
ERRCAL	7463#	8201	8247	8315	8455	8513									
ERREND	2903#	4551													
ERROR	1631#	3072	3546	3560	3575	3642	3666	3724	3742	3759	3816	3834	3851	5792	5793
	5794	5795	5796	5824	5825	5826	5827	5828	5863	5864	5865	5866	5867	5909	5910
	5911	5912	5913	5939	5940	5941	5942	5943	5965	5966	5967	5968	5969	6107	6144
	6145	6146	6147	6148	6467	6472	6598	6604	6680	6681	6853	8201	8247	8315	8456
	8513														
ERRTYP	2903#	4521													
ER. NDX	2903#	5786	5818	5857	5903	5933	5959	6138							
ESCAPE	1#	1731#	3522	3621	3704	3796	5785	5817	5856	5902	5932	5958	6105	6137	6456
	6470	6597	6602	6663											
MORETA	1795#	1848													
MORE.S	2903#	3184	3216	3251	3288	3322	3396	3435	3502	3604	3687	3779	3889	3947	4065
	4183	4290													
MSG	3174#	3176	3204#	3206	3238#	3240	3279#	3281	3312#	3314	3383#	3385	3424#	3426	3489#
	3492	3591#	3593	3676#	3678	3768#	3770	3876#	3878	3933#	3935	4003#	4005	4158#	4160
	4247#	4249													
MULT	1#	1731#													
NEWTST	1#	1731#	3174	3204	3238	3279	3312	3383	3424	3490	3591	3676	3768	3876	3933
	4003	4158	4247												
POP	1#	1731#	4760	5029	5268	5531	5577								
PUSH	1#	1731#	4722	4988	5248	5483	5549								
REPORT	2903#	3583	3586	3672	3765	3857									
RPO4.D	1552#	7463													
SAV. RH	2903#	3540	3554	3569	3636	3660	3718	3736	3753	3810	3828	3845			
SCOPE	1632#	3202	3236	3277	3310	3381	3422	3468	3576	3667	3760	3852	3931	4000	4146
	4244	4439													
SETTRA	5291#	5301	5302	5303	5304	5305	5306	5307	5308	5309					
SETUP	1#	1731#	2929												
SET. TN	2903#	3185	3217	3252	3289	3323	3397	3436	3503	3605	3688	3780	3890	3948	4066
	4184	4291													
SKIP	1#	1731#													
SLASH	1#	1731#													
SPACE	1731#														
STARS	1#	1731#	1795	2597	2802	3174	3183	3204	3215	3238	3250	3279	3287	3312	3321
	3383	3395	3424	3434	3490	3501	3591	3603	3676	3686	3768	3778	3876	3888	3933
	3946	4003	4064	4158	4182	4247	4289	4443	4505	4555	4773	4775	4799	4801	4850
	4897	4975	5043	5092	5112	5173	5229	5275	5310	5350	5368	5431	5455	5537	5584
	5661	5681	5751	5769	5800	5841	5881	5981	6033	6062	6084	6112	6155	6161	6181
	6220	6305	6324	6342	6364	6389	6483	6519	6613	6634	6695	6716	6771	6796	6860
	6919	7274	7294	7314	7348	7414	7462								
TRMTRP	5291#														
TYPBIN	1#	1731#													
TYPB2D	2903#	6250	6257	6261	6266	6273	6277	6289	6294	7119	7190				

ADC	5399	5564	5566	5568	5569	5571	5574	6212	6747						
ADD	2952	3113	3268	3358	3416	3456	3460	3464	3466	3927	3968	3977	3987	3991	4080
	4082	4092	4339	4350	4356	4365	4367	4377	4435	4436	4581	4582	4756	4882	4926
	4936	5008	5347	5398	5400	5509	5563	5565	5567	5570	5572	5573	5623	5631	5640
	5641	5791	5823	5862	5908	5938	5964	6079	6143	6211	6217	6316	6335	6440	6449
	6460	6514	6569	6585	6586	6652	6653	6664	6751	7005	7023	7379	7380	7405	7437
	7458	7869	7870	7952	8027	8083	8125	8445	8500	8540	8673	8707			
ASL	3307	4592	4600	4738	4740	4742	4744	4746	4748	5558	6027	6459	6584	6828	6829
	6830	7002	7003	7004	7008	7020	7021	7022	7026	7081	7269	7375	7377	7378	7431
	7433	7435	7724	7886	7888	7890	7892	7894	7896	8041	8155	8249	8291	8654	8671
	8692	8705													
ASLB	3016	3065	4577	4578	4579	4591	5013	8185	8222						
ASR	2976	2977	2978	2979	2980	2983	2985	3036	4343	4344	4345	4346	4347	4730	4731
	4732	4733	5339	6455	6581	6676	6914	6915	6916	7098	7107	7880	7881	7882	7883
	8043	8117	8118	8157	8251	8293	8656	8677	8694	8711					
BCC	3037	4580	4601	5014	5510	7181	8186								
BCS	2953	3066	4661	7099	7108	7432	7434	7436							
BEQ	3006	3044	3055	3337	3345	3351	3365	3371	3526	3539	3553	3635	3717	3735	3809
	3827	4088	4109	4122	4133	4135	4338	4373	4394	4407	4418	4420	4458	4490	4522
	4524	4528	4531	4545	4548	4584	4608	4617	4619	4621	4623	4625	4646	4651	4655
	4693	4953	5104	5140	5203	5207	5334	5445	5447	5527	5721	5783	5815	5836	5854
	5875	5897	5921	5930	5949	5955	5976	6020	6046	6098	6135	6195	6248	6255	6271
	6299	6315	6334	6445	6576	6662	6691	6782	6809	6855	6905	6971	7033	7039	7041
	7056	7058	7065	7083	7100	7109	7167	7171	7218	7222	7261	7263	7266	7271	7330
	7332	7334	7336	7399	7729	7738	7827	7905	7908	7992	7994	8026	8030	8052	8072
	8075	8241	8262	8299	8304	8310	8387	8399	8402	8460	8508	8530	8669	8691	
BGE	3270	3380	3421	4079	4084	4094	4211	4364	4369	4379	4432	5217	5522	6198	6201
	6757	7029	7077	7838	7951	8124	8129	8183	8471						
BGT	3053	3071	3118	3356	3376	3419	3453	3513	3563	3615	3654	3698	3747	3790	3839
	3930	3971	3980	3996	3999	4104	4115	4117	4208	4389	4400	4402	4480	4960	5022
	5333	5498	6022	6357	6360	6443	6631	6712	6732	6739	6753	7007	7025	7215	7788
	7795	7829	8114	8208	8231	8309	8356	8362	8411						
BHI	5205	5633	5973	6447	6608	6689	7288	7308	7404						
BIC	2984	3537	3551	3565	3633	3656	3715	3733	3749	3807	3825	3841	4340	4477	4729
	4751	4950	5074	5346	5482	5637	6049	6094	6456	7290	7310	7697	7761	7879	7898
	8084	8495													
BICB	7824														
BIS	4752	4955	4956	5016	5017	5402	5499	5517	5615	7030	7036	7044	7796	7807	7831
	7899	7906	7909	7935	8036	8078	8160	8172	8274	8285	8295	8311	8325	8389	8404
	8463	8558													
BISB	3014	6950	6955	7962											
BIT	3019	3054	3062	3185	3217	3252	3289	3323	3336	3397	3436	3503	3525	3538	3552
	3581	3605	3634	3688	3716	3734	3780	3808	3826	3890	3948	4066	4108	4121	4124
	4128	4134	4136	4140	4184	4200	4291	4309	4337	4393	4406	4409	4413	4419	4421
	4425	4521	4530	4537	4544	5206	5213	5720	5730	5740	5760	5835	5874	5920	5922
	5970	5975	6003	6006	6009	6012	6015	6018	6045	6238	6473	6475	6479	6605	6682
	6686	6781	7032	7082	7251	7260	7262	7265	7270	7728	7904	7907	8219	8240	8451
	8459	8507	8561												
BITB	8321	8417													
BLE	3013	3276	3309	3360	4697	6205	6208	6318	6337	6744	6762	8233	8239	8318	8544
BLO	2917	2922	5135	6052	6174	6385	7286	7306							
BLOS	5123	6215	6478	6685	7686	7690									
BLT	3649	3986	4890	4961	5005	5021	5395	5497	6730	6737	7235	7248	7834	8127	8143
	8198	8354	8490												
BMI	4589	5012	5189	5950	7113	7759	7840	8166	8195	8273	8288	8384	8458	8536	
BNE	2933	2949	3017	3020	3030	3063	3186	3218	3253	3290	3324	3398	3437	3504	3582

	3606	3646	3689	3781	3891	3949	4067	4125	4129	4137	4141	4185	4201	4292	4310
	4353	4410	4414	4422	4426	4492	4494	4538	4603	4611	4628	4672	4674	4676	4678
	4681	4683	4700	4707	4879	4886	4951	5010	5076	5081	5089	5109	5127	5129	5145
	5149	5159	5214	5406	5514	5562	5644	5731	5741	5761	5832	5871	5915	5917	5919
	5923	5971	6005	6008	6011	6014	6017	6056	6096	6239	6407	6409	6411	6413	6415
	6417	6419	6421	6423	6425	6427	6429	6431	6433	6435	6437	6439	6457	6463	6474
	6476	6480	6494	6515	6531	6538	6540	6542	6544	6546	6548	6550	6552	6554	6556
	6558	6560	6562	6564	6566	6568	6570	6592	6606	6626	6649	6671	6678	6683	6687
	6748	6902	6907	6909	6952	6967	6969	6976	6980	6984	6988	6992	7011	7043	7047
	7050	7052	7061	7068	7132	7135	7138	7140	7169	7220	7252	7402	7452	7457	7698
	7740	7762	7790	7801	7804	7874	7902	7934	7968	7975	7982	7996	8006	8016	8063
	8085	8111	8200	8206	8220	8223	8227	8323	8348	8418	8420	8452	8488	8534	8562
BPL	8638	8643	8675	8709											
	4542	4873	4894	4949	4996	5026	5491	5516	5612	5784	5816	5855	5898	5931	5956
BR	6136	6287	7753	8038	8168	8213	8260	8281	8327	8425					
	2908	2923	2925	2955	2972	2988	3000	3008	3024	3032	3046	3050	3057	3067	3073
	3090	3099	3467	3516	3517	3564	3585	3618	3619	3651	3655	3701	3702	3748	3793
	3794	3840	3988	4101	4106	4144	4145	4386	4391	4429	4430	4454	4460	4468	4485
	4596	4610	4613	4626	4632	4638	4679	4685	4688	4695	4709	4779	4785	4804	4808
	4814	4819	4825	4831	4875	4892	4927	4942	4963	5007	5024	5138	5147	5153	5155
	5191	5197	5200	5209	5212	5348	5397	5501	5503	5506	5636	5676	5709	5715	5724
	5764	5781	5813	5830	5852	5869	5895	5899	6057	6133	6150	6301	6469	6578	6601
	6610	6674	6707	6735	6742	6755	6760	6765	6784	6790	6791	6793	6815	6816	6817
	6822	6823	6824	6833	6834	6835	6837	6839	6900	6903	6929	6954	6961	6978	6982
	6986	6990	6994	6996	7000	7014	7018	7034	7045	7053	7074	7085	7097	7102	7104
	7111	7133	7151	7172	7187	7216	7223	7227	7231	7237	7253	7338	7400	7429	7439
	7440	7695	7706	7731	7764	7793	7797	7799	7806	7814	7832	7836	7842	7843	7845
	7851	7923	7963	7973	7980	7990	8014	8028	8050	8145	8171	8176	8192	8229	8243
	8248	8275	8286	8294	8301	8312	8337	8340	8358	8397	8415	8431	8462	8497	8538
	8564														
CLC	5505	6733	6740	6745	6758										
CLR	2909	2911	2926	2931	2941	2942	2947	2960	2997	3009	3010	3018	3027	3028	3060
	3104	3335	3341	3357	3377	3523	3533	3547	3566	3624	3628	3657	3710	3725	3728
	3750	3802	3817	3820	3842	3916	3965	3973	3981	4075	4076	4089	4099	4107	4203
	4313	4348	4349	4360	4361	4374	4384	4392	4440	4473	4474	4475	4594	4652	4737
	4940	4999	5002	5061	5101	5102	5120	5143	5211	5223	5329	5391	5487	5518	5557
	5618	5626	5635	5639	5671	5673	5702	5705	5778	5810	5849	5891	5892	5925	6002
	6050	6122	6123	6127	6128	6130	6151	6152	6172	6300	6352	6400	6402	6534	6660
	6810	6856	6857	6927	6953	6972	6973	6974	7072	7106	7183	7365	7366	7367	7427
	7442	7684	7691	7726	7785	7872	7885	8064	8069	8070	8087	8159	8180	8215	8218
CLRB	8351	8352	8394	8395	8413	8531	8537	8630	8631	8632	8633	8689			
	3905	3925	5028	5150	5160	5210	5407	5524	7705	7765	7810	7998	8060	8061	8080
CMP	8081	8130	8148	8153	8296	8300	8319	8320	8364	8392	8393	8405	8406	8653	
	2916	2921	2932	2948	3269	3275	3308	3355	3359	3375	3379	3418	3420	3452	3648
	3653	3929	3970	3979	3998	4103	4114	4116	4210	4388	4399	4401	4431	4491	4493
	4675	4677	4682	5020	5075	5080	5088	5108	5122	5134	5198	5216	5401	5632	5643
	5677	5710	5716	6058	6095	6097	6173	6197	6200	6204	6207	6214	6314	6317	6333
	6336	6359	6384	6406	6408	6410	6412	6414	6416	6418	6420	6422	6424	6426	6428
	6430	6432	6434	6436	6442	6688	6743	6756	6761	6852	7006	7024	7027	7076	7214
	7234	7247	7403	7685	7689	7737	7739	8025	8062	8074	8109	8126	8128	8226	8361
CMPB	8386	8401	8410	8419	8453	8533	8543	8674	8708						
	3995	4207	4616	4618	4620	4622	4624	4671	4673	4680	4885	5126	5144	5148	5158
	5204	5446	5831	5870	5914	5916	5918	5972	6021	6446	6477	6493	6530	6575	6607
	6625	6648	6670	6684	6731	6738	6906	6966	6968	6975	6979	6983	6987	6991	7010
	7038	7042	7046	7055	7137	7168	7219	7285	7287	7305	7307	7331	7333	7335	7401
	7451	7800	7833	7873	7933	7967	7974	7981	7991	7993	7995	8005	8015	8029	8113

COM	8489	8668													
COMB	3021	3338	4081	4120	4366	4405	4602	4656	4693	7760					
DEC	6468	6600	6679												
	3117	3562	3652	3746	3838	3985	4083	4093	4352	4368	4378	4478	4576	4692	4696
	5105	5133	5332	5404	5448	5513	5525	6055	6356	6438	6630	6711	6941	8197	8207
DECB	8469	8637	8642												
EMT	4889	4948	4959	5492	8704										
HALT	1631														
INC	1614	4543	4874	5659											
	2981	3015	3038	3049	3064	3354	3374	3647	3928	3969	3978	3997	4209	4351	4434
	4476	4533	4954	4962	5006	5085	5087	5107	5215	5335	5396	5502	5561	5628	5675
	6023	6024	6025	6026	6202	6209	6213	6288	6358	6441	6752	6763	6998	7001	7012
	7016	7019	7035	7048	7054	7063	7075	7213	7233	7344	7430	7453	7696	8082	8184
INCB	8221	8359	8409	8466											
IOT	3932	4206	4527	5220	8112	8670									
JMP	1632														
	1608	1620	1622	1624	3035	3187	3219	3254	3291	3325	3399	3438	3505	3514	3607
	3616	3690	3699	3782	3791	3892	3950	4068	4186	4293	4311	4433	4437	4500	5079
	5910	6481	6611	6858	7059	7062	7066	7069	7079	7136	7141	7242	7250	7942	8306
JSR	8307	8316	8330												
	2958	2959	2970	2991	2992	2993	2994	3007	3078	3082	3084	3088	3200	3201	3234
	3235	3267	3273	3303	3306	3343	3346	3349	3352	3363	3366	3369	3372	3413	3415
	3455	3457	3459	3461	3463	3465	3515	3530	3542	3556	3561	3571	3579	3583	3587
	3617	3625	3638	3643	3644	3662	3670	3672	3700	3707	3720	3726	3727	3738	3744
	3745	3755	3763	3765	3792	3799	3812	3818	3819	3830	3836	3837	3847	3855	3857
	3901	3908	3910	3911	3913	3914	3921	3926	3959	3967	3976	3990	3994	4105	4123
	4127	4131	4139	4142	4143	4202	4205	4216	4222	4223	4224	4227	4228	4233	4234
	4237	4238	4243	4390	4408	4412	4416	4424	4427	4428	4496	4539	4566	4710	4755
	4884	4891	5078	5364	5708	5714	5748	5779	5790	5811	5822	5833	5837	5850	5861
	5872	5876	5893	5907	5926	5937	5945	5951	5963	5977	6047	6131	6142	6251	6252
	6258	6259	6262	6263	6267	6268	6274	6275	6278	6279	6285	6290	6291	6295	6296
	6351	6361	6372	6386	6398	6450	6491	6516	6528	6572	6629	6657	6708	6725	6789
	6814	6821	6832	6884	6892	6926	6934	6944	6956	6995	6999	7013	7017	7071	7078
	7120	7121	7124	7144	7159	7174	7182	7191	7192	7198	7206	7225	7226	7229	7230
	7241	7245	7249	7254	7272	7337	7368	7381	7390	7428	7438	7443	7680	7681	7694
	7702	7721	7730	7732	7742	7746	7749	7755	7767	7783	7792	7798	7805	7809	7813
	7822	7825	7830	7835	7841	7844	7846	7848	7850	7864	7876	7922	7926	7930	7938
	7946	7955	7959	7970	7977	7983	7987	8001	8007	8011	8021	8032	8035	8039	8047
	8054	8056	8059	8067	8076	8077	8088	8090	8091	8106	8120	8134	8140	8144	8146
	8147	8162	8169	8170	8173	8174	8189	8202	8209	8234	8235	8236	8244	8252	8255
	8263	8267	8269	8276	8283	8297	8305	8314	8324	8328	8329	8336	8350	8357	8363
	8377	8380	8388	8390	8396	8403	8414	8421	8427	8430	8491	8504	8510	8526	8545
MOV	8628	8644													
	2907	2910	2912	2913	2914	2915	2918	2919	2920	2924	2927	2929	2930	2934	2935
	2936	2937	2938	2939	2940	2944	2945	2946	2950	2951	2975	2982	2995	2998	2999
	3001	3002	3003	3004	3011	3029	3039	3051	3061	3069	3074	3075	3076	3077	3079
	3081	3083	3085	3087	3093	3102	3103	3114	3188	3189	3180	3191	3193	3194	3197
	3198	3199	3220	3221	3222	3223	3225	3230	3231	3232	3233	3255	3256	3257	3258
	3260	3263	3264	3265	3271	3292	3293	3294	3295	3297	3300	3301	3302	3304	3305
	3326	3327	3328	3329	3331	3334	3339	3340	3342	3348	3361	3362	3368	3400	3401
	3402	3403	3405	3408	3409	3410	3411	3412	3414	3439	3440	3441	3442	3444	3447
	3448	3449	3450	3454	3506	3507	3508	3509	3511	3518	3519	3521	3522	3524	3527
	3528	3529	3531	3532	3534	3535	3541	3543	3544	3548	3549	3555	3557	3558	3567
	3570	3572	3573	3577	3578	3580	3608	3609	3610	3611	3613	3620	3621	3622	3623
	3626	3627	3629	3630	3631	3637	3639	3640	3650	3658	3661	3663	3664	3668	3669
	3671	3691	3692	3693	3694	3696	3703	3704	3705	3706	3708	3709	3711	3712	3713

CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

3719	3721	3722	3729	3730	3731	3737	3739	3740	3743	3751	3754	3756	3757	3761
3762	3764	3783	3784	3785	3786	3788	3795	3796	3797	3798	3800	3801	3803	3804
3805	3811	3813	3814	3821	3822	3823	3829	3831	3832	3835	3843	3846	3848	3849
3853	3854	3856	3893	3894	3895	3896	3898	3899	3900	3902	3903	3906	3907	3909
3912	3915	3917	3918	3919	3922	3923	3924	3951	3952	3953	3954	3956	3957	3958
3960	3961	3963	3964	3972	3974	3983	3984	3989	3993	4069	4070	4071	4072	4074
4077	4086	4090	4097	4098	4100	4110	4111	4113	4118	4119	4126	4130	4138	4187
4188	4189	4190	4192	4193	4194	4195	4196	4197	4198	4199	4212	4213	4214	4215
4217	4218	4219	4220	4221	4225	4226	4229	4230	4231	4232	4235	4236	4239	4240
4241	4242	4294	4295	4296	4297	4299	4314	4315	4316	4317	4318	4319	4320	4323
4324	4325	4326	4327	4328	4329	4330	4331	4334	4335	4336	4341	4355	4357	4359
4362	4371	4375	4382	4383	4385	4395	4396	4398	4403	4404	4411	4415	4423	4463
4471	4481	4489	4525	4526	4529	4534	4546	4549	4551	4552	4568	4569	4570	4571
4572	4573	4575	4583	4590	4593	4599	4629	4630	4635	4641	4643	4645	4650	4653
4654	4662	4690	4702	4723	4724	4725	4726	4727	4736	4753	4754	4757	4760	4761
4762	4763	4764	4789	4822	4828	4834	4836	4876	4877	4881	4887	4923	4931	4932
4933	4939	4946	4964	4965	4966	4967	4968	4989	4990	4991	4992	4993	4994	4995
5000	5003	5023	5029	5030	5031	5032	5033	5035	5036	5062	5063	5064	5065	5067
5090	5099	5100	5110	5119	5121	5132	5163	5164	5165	5166	5193	5194	5196	5199
5208	5218	5219	5221	5222	5225	5226	5248	5249	5250	5251	5252	5253	5254	5255
5256	5257	5264	5265	5266	5267	5268	5269	5270	5271	5272	5273	5284	5285	5288
5323	5324	5325	5326	5327	5328	5331	5336	5362	5363	5365	5383	5384	5385	5386
5387	5388	5389	5390	5442	5443	5449	5452	5453	5481	5483	5484	5485	5486	5488
5489	5490	5496	5500	5508	5511	5519	5529	5530	5531	5532	5533	5534	5535	5550
5551	5552	5553	5554	5555	5556	5575	5576	5577	5578	5579	5580	5601	5602	5603
5604	5605	5606	5607	5608	5609	5613	5616	5617	5619	5620	5621	5622	5625	5627
5629	5634	5638	5647	5649	5649	5650	5651	5652	5653	5654	5655	5672	5678	5699
5700	5701	5703	5704	5707	5711	5713	5717	5718	5719	5722	5723	5725	5726	5732
5733	5734	5735	5742	5743	5744	5747	5762	5763	5765	5766	5785	5786	5789	5797
5817	5818	5821	5829	5856	5857	5860	5868	5902	5903	5906	5932	5933	5936	5957
5958	5959	5962	5974	5998	5999	6000	6001	6028	6029	6030	6044	6048	6053	6072
6073	6074	6075	6076	6077	6080	6081	6092	6093	6099	6104	6105	6108	6109	6124
6125	6126	6129	6137	6138	6141	6170	6171	6175	6176	6177	6193	6196	6199	6206
6216	6237	6240	6241	6242	6243	6244	6250	6257	6261	6266	6273	6277	6282	6283
6284	6294	6298	6319	6338	6353	6354	6355	6373	6374	6375	6376	6377	6378	6379
6380	6381	6382	6383	6401	6403	6404	6448	6452	6454	6458	6461	6464	6466	6471
6492	6495	6496	6497	6498	6499	6500	6501	6502	6503	6504	6505	6506	6507	6508
6509	6510	6511	6512	6513	6529	6532	6533	6535	6574	6582	6587	6588	6589	6590
6593	6596	6597	6599	6603	6609	6624	6627	6628	6644	6645	6646	6647	6650	6651
6654	6655	6656	6658	6659	6663	6665	6666	6667	6668	6669	6692	6705	6706	6709
6710	6713	6728	6749	6768	6787	6788	6792	6811	6812	6813	6818	6819	6820	6825
6826	6827	6836	6838	6840	6841	6842	6844	6845	6846	6847	6848	6849	6851	6877
6880	6883	6899	6912	6933	6943	6965	6977	6981	6985	6989	6993	6997	7009	7015
7080	7088	7094	7095	7096	7103	7105	7115	7119	7123	7143	7150	7155	7158	7179
7184	7190	7194	7197	7211	7224	7228	7232	7236	7239	7243	7255	7256	7257	7258
7259	7264	7345	7362	7363	7364	7376	7407	7409	7410	7411	7426	7441	7459	7460
7678	7679	7682	7683	7687	7688	7692	7693	7699	7700	7701	7703	7722	7723	7735
7736	7741	7745	7752	7780	7781	7784	7791	7811	7823	7847	7865	7866	7867	7868
7871	7875	7877	7884	7900	7903	7911	7924	7925	7929	7937	7941	7943	7944	7945
7953	7958	7964	7965	7969	7976	8000	8017	8024	8031	8042	8046	8053	8055	8065
8068	8073	8079	8086	8089	8104	8105	8119	8141	8142	8154	8156	8158	8161	8175
8181	8187	8188	8190	8191	8193	8196	8204	8217	8242	8246	8250	8253	8254	8258
8266	8268	8279	8284	8292	8313	8331	8332	8333	8334	8335	8365	8375	8376	8378
8379	8385	8391	8400	8407	8412	8416	8428	8442	8443	8444	8446	8449	8450	8465
8472	8473	8485	8486	8487	8498	8501	8509	8515	8527	8529	8532	8539	8541	8554
8555	8556	8557	8566	8629	8634	8635	8636	8639	8640	8641	8655	8672	8676	8693

MOVE	8706	8710	8734	8735	8736	8737	8738	8739	8752	8754	8755	8756	8757	8758	
	2943	3080	3086	3195	3196	3226	3227	3228	3229	3261	3262	3298	3299	3332	3333
	3406	3407	3445	3446	3520	3904	3920	3962	3966	3975	3982	4096	4112	4381	4397
	4536	4565	4574	4658	4659	4878	4895	4924	4925	4928	4929	4930	4934	4937	4938
	4957	4998	5001	5015	5018	5027	5073	5086	5106	5125	5130	5136	5141	5156	5224
	5287	5330	5403	5787	5788	5819	5820	5858	5859	5904	5905	5924	5934	5935	5944
	5960	5961	6100	6101	6102	6103	6139	6140	6465	6594	6595	6726	6727	6766	6767
	7289	7309	7725	7727	7754	7763	7782	7786	7802	7949	7954	7966	7986	7997	7999
	8010	8019	8020	8044	8115	8139	8194	8205	8216	8282	8289	8290	8349	8467	8496
	8528	8560	8565												
NEG	4085	4095	4370	4390	4935	4997	5493	5494	5504	5523	5528	6754			
NOP	4497	4498	4499	5928	5947	5953									
RESET	2928	4495													
ROL	3105	3106	3107	3108	3109	3110	3111	3112	4739	4741	4743	4745	4747	4749	4941
	4943	4944	4945	4947	5188	5507	5512	5559	5560	6286	7758	7887	7889	7891	7893
	7895	7897	8165	8259	8272	8280									
ROLB	8212														
ROR	4660	5340	5341	5342	5343	5344	5345	6746	6759	7180					
RORB	6734	6741													
RTI	4553	4765	4790	4837	4883	4969	5037	5084	5091	5111	5167	5227	5258	5274	5536
	5749	6159	8149												
RTS	4711	4896	5068	5289	5338	5366	5409	5454	5581	5656	5679	5727	5737	5745	5767
	5798	5839	5878	5979	6031	6059	6082	6110	6153	6179	6218	6303	6321	6340	6362
	6387	6451	6517	6573	6632	6693	6714	6769	6794	6850	6917	6957	7238	7273	7292
	7312	7346	7412	7461	7704	7768	7812	7849	8004	8040	8045	8066	8092	8133	8135
	8203	8214	8225	8338	8366	8429	8454	8474	8499	8516	8518	8546	8567	8645	8657
	8679	8695	8712	8740	8759										
SBC	5393	5495													
SDB	5624														
SUB	3274	3378	3417	3451	3458	3462	4078	4091	4358	4363	4376	4535	4567	4588	4782
	4811	5004	5392	5394	5645	5646	6051	6399	6453	6537	6539	6541	6543	6545	6547
	6549	6551	6553	6555	6557	6559	6561	6563	6565	6567	6577	6579	6583	6661	6672
	6673	6675	7950	8123	8355										
SWAB	4342	4728	6750	6764	6831	6913	7878	8116	8464	8494	8559				
TRAP	5291	5301	5302	5303	5304	5305	5306	5307	5308	5309					
TST	2996	3005	3043	3052	3344	3350	3364	3370	3512	3568	3614	3645	3659	3697	3752
	3789	3844	4087	4102	4132	4354	4372	4387	4417	4457	4523	4541	4547	4607	4627
	4706	4880	4952	5009	5019	5066	5083	5103	5128	5139	5162	5195	5286	5515	5520
	5521	5526	5614	5630	5642	5674	5706	5712	5782	5814	5853	5896	5929	5948	5954
	6054	6106	6134	6149	6194	6203	6210	6247	6254	6270	6320	6339	6444	6462	6571
	6591	6677	6690	6808	6843	6854	6897	6898	6910	6911	7031	7040	7051	7060	7067
	7101	7110	7112	7212	7240	7244	7246	7267	7268	7291	7311	7339	7340	7341	7342
	7343	7388	7389	7397	7406	7408	7450	7454	7455	7766	7808	7826	7837	7839	7901
	8037	8131	8132	8167	8224	8228	8302	8326	8339	8347	8353	8360	8408	8426	8457
	8461	8514	8517	8542	8563	8678	8753								
TSTB	3012	3070	4872	4893	5011	5025	5202	5444	5610	6725	6736	6901	6904	6908	6951
	6970	7049	7057	7084	7131	7134	7139	7166	7170	7217	7221	7329	7398	7456	7787
	7789	7794	7803	7828	8051	8071	8182	8199	8230	8232	8238	8261	8287	8298	8303
	8308	8317	8383	8398	8424	8535	8690								
WAIT	3536	3550	3632	3714	3732	3806	3824	6078							
.ASCII	1845	1846													
.ASCIZ	1844	1847	2957	2974	2990	3026	3034	3048	3059	3092	3101	4456	4462	4470	4487
	4598	4615	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	5170	5172	6786
	6931	6963	7087	7189	8764	8765	8766	8767	8768	8769	8770	8771	8772	8773	8774
	8775	8779	8781	8783	8786	8791	8800	8805	8812	8815	8822	8826	8828	8830	8832
	8833	8837	8841	8844	8846	8850	8857	8863	8868	8873	8882	8890	8895	8899	8903

	8908	8915	8918	8926	8932	8937	8940	8944	8948	8954	8960	8966	8973	8976	8986
	8989	9000	9005	9012	9015	9018	9028	9036	9043	9054	9062	9068	9073	9082	9086
	9094	9098													
.BLKB	5050	5171	5349	5430											
.BLKW	5042	8613	8614	8615	8616	8617	8618	8619	8620						
.BYTE	1809	1810	1815	1816	1827	1828	1829	1830	1916	1918	2477	2478	2479	2480	2484
	2486	2501	2502	2503	2504	2508	2510	2525	2526	2527	2528	2532	2534	2549	2550
	2551	2552	2556	2558	3041	3042	3095	3096	4465	4466	4501	4970	4971	4972	4973
	5168	5169	7091	7092	7488	7489	7490	7491	7492	7493	7494	7495	7502	7503	7504
	7505	7506	7507	7508	7509	7542	7548	7555	7556	7557	7558	7559	7560	7561	7562
	7567	7568	7569	7570	7571	7572	7573	7574	7614	7615	7616	7617	7618	7619	7620
	7621	8570	8571	8572	8573	8574	8575	8576	8577	9168	9169	9172	9173	9176	9177
	9180	9181	9184	9185	9188	9189	9192	9193	9195	9196	9199	9200	9202	9203	9206
	9207	9210	9211	9214	9215	9217	9218	9221	9222	9225	9226	9229	9230	9233	9234
	9236	9237	9240	9241	9243	9244	9246	9247	9250	9251	9253	9254	9256	9257	
.ENABL	1	1552													
.END	9262														
.ENDC	1558	1593	1595	1596	1597	1614	1631	1717	1731	1744	1755	1766	1796	1804	1806
	1831	1839	1842	1843	1844	1845	2597	2598	2804	2907	2934	2935	2937	2939	2941
	2942	2944	2946	2957	2974	2990	3026	3034	3042	3043	3048	3059	3081	3083	3087
	3092	3096	3097	3101	3127	3154	3162	3172	3175	3176	3183	3184	3188	3190	3194
	3205	3206	3215	3216	3220	3222	3226	3239	3240	3250	3251	3255	3257	3261	3280
	3281	3287	3288	3292	3294	3298	3313	3314	3321	3322	3326	3328	3332	3384	3385
	3395	3396	3400	3402	3406	3425	3426	3434	3435	3439	3441	3445	3476	3488	3491
	3492	3501	3502	3506	3508	3512	3523	3592	3593	3603	3604	3608	3610	3614	3622
	3677	3678	3686	3687	3691	3693	3697	3705	3769	3770	3778	3779	3783	3785	3789
	3797	3866	3874	3877	3878	3888	3889	3893	3895	3899	3934	3935	3946	3947	3951
	3953	3957	4004	4005	4064	4065	4069	4071	4075	4159	4160	4182	4183	4187	4189
	4193	4248	4249	4289	4290	4294	4296	4300	4444	4447	4449	4451	4456	4462	4466
	4467	4470	4474	4480	4483	4484	4487	4489	4501	4502	4506	4512	4527	4534	4540
	4541	4551	4554	4556	4598	4615	4634	4640	4774	4776	4781	4787	4800	4802	4806
	4810	4816	4821	4827	4833	4851	4898	4976	5044	5080	5093	5112	5113	5121	5123
	5154	5171	5172	5173	5174	5180	5184	5188	5190	5201	5202	5204	5206	5213	5215
	5220	5225	5228	5229	5230	5276	5285	5288	5290	5300	5301	5302	5303	5304	5305
	5306	5307	5308	5309	5311	5351	5369	5432	5456	5538	5585	5618	5659	5662	5682
	5732	5770	5786	5797	5801	5818	5829	5842	5857	5868	5882	5903	5914	5933	5944
	5959	5970	5982	6034	6063	6085	6106	6113	6138	6149	6156	6162	6182	6221	6306
	6325	6343	6365	6390	6467	6472	6484	6520	6598	6604	6614	6635	6664	6696	6717
	6772	6786	6797	6861	6920	6931	6963	7087	7092	7093	7189	7275	7295	7315	7349
	7415	7463													
.EQUIV	1631	1632	1634	1649	1650	1679	1680	1681	1682	1683	1684	1685	1686	1687	1688
	1707	1708	1709	1710	1711	1712	1713	1714	1715	1716					
.EVEN	2957	2974	2990	3026	3034	3048	3059	3092	3101	4456	4462	4470	4487	4502	4598
	4615	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	6786	6931	6963	7087
	7189	9108	9259												
.IF	1554	1593	1594	1595	1596	1614	1629	1689	1717	1744	1755	1766	1795	1801	1805
	1831	1839	1842	1843	1844	1848	2597	2802	2907	2929	2934	2935	2937	2939	2941
	2942	2944	2956	2973	2989	3025	3033	3041	3042	3047	3058	3080	3083	3087	3091
	3095	3096	3100	3124	3151	3159	3169	3174	3176	3183	3187	3189	3194	3204	3206
	3215	3219	3221	3226	3238	3240	3250	3254	3256	3261	3279	3281	3287	3291	3293
	3298	3312	3314	3321	3325	3327	3332	3383	3385	3395	3399	3401	3406	3424	3426
	3434	3438	3440	3445	3473	3485	3490	3492	3501	3505	3507	3512	3522	3591	3593
	3603	3607	3609	3614	3621	3676	3678	3686	3690	3692	3697	3704	3768	3770	3778
	3782	3784	3789	3796	3863	3871	3876	3878	3888	3892	3894	3899	3933	3935	3946
	3950	3952	3957	4003	4005	4064	4068	4070	4075	4158	4160	4182	4186	4188	4193
	4247	4249	4289	4293	4295	4300	4443	4447	4448	4449	4450	4451	4453	4455	4461

	4465	4466	4469	4479	4482	4484	4486	4489	4500	4501	4505	4511	4521	4530	4537
	4539	4540	4541	4544	4551	4554	4555	4597	4614	4633	4639	4773	4775	4780	4786
	4799	4801	4805	4809	4815	4820	4826	4832	4850	4897	4975	5043	5047	5075	5092
	5112	5120	5122	5127	5170	5171	5172	5173	5179	5184	5188	5200	5202	5203	5204
	5213	5215	5222	5227	5228	5229	5275	5284	5288	5290	5291	5301	5302	5303	5304
	5305	5306	5307	5308	5309	5310	5350	5368	5431	5455	5537	5594	5591	5616	5661
	5681	5751	5769	5785	5796	5800	5817	5828	5841	5856	5867	5881	5902	5913	5932
	5943	5958	5969	5981	6033	6062	6084	6105	6112	6137	6148	6155	6161	6181	6220
	6305	6324	6342	6364	6389	6466	6471	6483	6519	6597	6603	6613	6634	6663	6695
	6716	6771	6785	6796	6860	6919	6930	6962	7086	7091	7092	7188	7274	7294	7314
.IFF	7348	7414	7462	7463											
	1593	1595	1596	1629	1796	1805	1831	2598	2802	2934	3042	3096	3174	3175	3176
	3184	3188	3190	3194	3204	3205	3206	3216	3220	3222	3226	3238	3239	3240	3251
	3255	3257	3261	3279	3280	3281	3288	3292	3294	3298	3312	3313	3314	3322	3326
	3328	3332	3383	3384	3385	3396	3400	3402	3406	3424	3425	3426	3435	3439	3441
	3445	3490	3491	3492	3502	3506	3507	3512	3522	3591	3592	3593	3604	3608	3609
	3614	3621	3676	3677	3678	3687	3691	3692	3697	3704	3768	3769	3770	3779	3783
	3784	3789	3796	3876	3877	3878	3889	3893	3895	3899	3933	3934	3935	3947	3951
	3953	3957	4003	4004	4005	4065	4069	4071	4075	4158	4159	4160	4183	4187	4188
	4193	4247	4248	4249	4290	4293	4296	4300	4444	4450	4453	4466	4479	4482	4501
	4506	4511	4530	4551	4556	4774	4776	4800	4802	4851	4898	4976	5044	5093	5112
	5113	5122	5154	5171	5174	5201	5202	5204	5228	5230	5276	5285	5311	5351	5369
	5432	5456	5538	5585	5600	5647	5662	5682	5752	5770	5785	5797	5801	5817	5829
	5842	5856	5868	5882	5902	5914	5932	5944	5958	5969	5982	6034	6063	6085	6105
	6113	6137	6149	6156	6162	6182	6221	6306	6325	6343	6365	6390	6466	6471	6484
	6520	6597	6603	6614	6635	6663	6696	6717	6772	6797	6861	6920	7092	7275	7295
.IFT	7315	7349	7415	7463											
	2957	2974	2990	3026	3034	3048	3059	3092	3101	4456	4462	4470	4487	4540	4598
	4615	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	5099	5212	5603	5609
.IFTF	5651	5658	6786	6931	6963	7087	7189								
	2957	2974	2990	3026	3034	3048	3059	3092	3101	4456	4462	4470	4487	4539	4598
	4615	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	5093	5210	5600	5605
	5647	5654	6786	6931	6963	7087	7189								
.IIF	1553	1558	1563	1590	1591	1592	1593	1597	1598	1599	1600	1601	1602	1603	1604
	1614	1848	2935	2937	2941	2942	2944	2945	3040	3094	4449	4464	4472	4474	4475
	4501	4502	4512	4513	4514	4515	4516	4554	4691	4897	5047	5052	5162	5172	5173
	5180	5181	5182	5183	5184	5211	5212	5225	5228	5229	5300	5301	5302	5303	5304
	5305	5306	5307	5308	5309	6250	6257	6261	6266	6273	6277	6290	6294	6881	7089
	7119	7156	7190	7195											
.IRP	1848	2907	3174	3204	3238	3279	3312	3383	3424	3490	3591	3676	3768	3876	3933
	4003	4158	4247	4453	4521	4551	4723	4760	4989	5029	5248	5268	5483	5531	5550
	5577	6405	6498	6536											
.LIST	1	1552	1614	1731	1831	1833	1834	1835	1836	1837	1838	1839	1840	1841	1842
	2802	2803	2804	2903	2907	2957	2974	2990	3026	3034	3048	3059	3092	3101	3123
	3155	3158	3173	3174	3194	3204	3226	3238	3261	3279	3298	3312	3332	3383	3406
	3424	3445	3472	3489	3490	3512	3591	3614	3676	3697	3768	3789	3862	3875	3876
	3899	3933	3957	4003	4075	4158	4193	4247	4300	4456	4462	4470	4487	4598	4615
	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	5112	5184	5290	5291	5300
	5301	5302	5303	5304	5305	5306	5307	5308	5309	5310	6786	6931	6963	7087	7189
.MACRO	1	1552	1597	1795	2903	3174	3204	3238	3279	3312	3383	3424	3489	3591	3676
	3768	3876	3933	4003	4158	4247	5291	7463							
.MCALL	1552	1731													
.NLIST	1	1552	1614	1731	1831	1833	1834	1835	1836	1837	1838	1839	1840	1841	1842
	2802	2803	2804	2903	2907	2957	2974	2990	3026	3034	3048	3059	3092	3101	3123
	3155	3158	3173	3174	3194	3204	3226	3238	3261	3279	3298	3312	3332	3383	3406
	3424	3445	3472	3489	3490	3512	3591	3614	3676	3697	3768	3789	3862	3875	3876

	3899	3933	3957	4003	4075	4158	4193	4247	4300	4456	4462	4470	4487	4598	4615
	4634	4640	4781	4787	4806	4810	4816	4821	4827	4833	5112	5184	5290	5291	5300
	5301	5302	5303	5304	5305	5306	5307	5308	5309	5310	6786	6931	6963	7087	7189
.PAGE	1731	1767	1795	2597	2903	3120	3469	3859	4001	4153	4245	4441	4503	7462	
.REM	1														
.REPT	1614	1833	1839	2802	3124	3151	3159	3169	3473	3485	3863	3671			
.SRTTL	1564	1586	1608	1615	1627	1732	1797	2599	2904	3121	3156	3174	3204	3238	3279
	3312	3383	3424	3470	3490	3591	3676	3768	3860	3876	3933	4001	4003	4156	4158
	4245	4247	4445	4504	4507	4556	4852	4899	4977	5045	5175	5231	5277	5292	5312
	5352	5370	5433	5457	5539	5586	5662	5682	5752	5770	5801	5842	5882	5982	6034
	6063	6085	6113	6156	6162	6182	6221	6306	6325	6343	6365	6390	6484	6520	6614
	6635	6696	6717	6772	6797	6861	6920	7275	7295	7315	7349	7415	7464	8762	8848
	8924	8984	9110	9165											
.TITLE	1553														
.WORD	1614	1808	1811	1812	1813	1814	1817	1818	1819	1820	1821	1822	1831	1833	1834
	1835	1836	1837	1838	1839	1840	1841	1862	1863	1864	1865	1866	1867	1868	1870
	1871	1872	1873	1874	1875	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886
	1887	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1901	1902	1903
	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1921	1922	1923
	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935	1936	1940	1941
	1942	1943	1944	1945	1946	1947	1948	1949	1950	1953	1954	1955	1956	1957	1958
	1959	1960	1961	1962	1963	1964	1965	1966	1967	1970	1971	1972	1973	1974	1975
	1976	1977	1978	1979	1982	1983	1984	1985	1986	1987	1988	1989	1990	1993	1994
	1995	1996	1997	1998	1999	2000	2001	2002	2003	2006	2008	2011	2014	2017	2020
	2023	2026	2028	2029	2030	2031	2033	2035	2039	2045	2046	2047	2048	2049	2052
	2053	2054	2055	2056	2057	2058	2059	2060	2063	2064	2065	2066	2067	2068	2069
	2072	2073	2074	2075	2076	2077	2078	2081	2082	2083	2084	2085	2086	2087	2090
	2091	2092	2093	2094	2095	2096	2099	2100	2101	2102	2103	2104	2105	2108	2109
	2110	2111	2112	2115	2116	2119	2120	2123	2124	2127	2128	2129	2130	2133	2134
	2135	2136	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2152	2153
	2154	2155	2159	2160	2161	2162	2164	2165	2166	2167	2169	2170	2171	2172	2174
	2175	2176	2177	2179	2180	2181	2182	2184	2185	2186	2187	2188	2189	2190	2191
	2192	2193	2194	2195	2196	2197	2198	2199	2203	2204	2205	2206	2207	2208	2209
	2210	2211	2212	2213	2214	2215	2216	2217	2218	2220	2221	2222	2223	2224	2225
	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2237	2238	2239	2240	2241
	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2254	2255	2256	2257
	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2271	2272	2273
	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2288	2289
	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2305
	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319	2320
	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335	2336
	2337	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352
	2353	2354	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367	2368
	2369	2370	2371	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383	2384
	2385	2386	2387	2388	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399	2400
	2401	2402	2403	2404	2405	2407	2408	2409	2410	2411	2412	2413	2414	2415	2416
	2417	2418	2419	2420	2421	2422	2424	2425	2426	2427	2428	2429	2430	2431	2432
	2433	2434	2435	2436	2437	2438	2439	2441	2442	2443	2444	2445	2446	2447	2448
	2449	2450	2451	2452	2453	2454	2455	2456	2458	2459	2460	2461	2462	2463	2464
	2465	2466	2467	2468	2469	2470	2471	2472	2473	2481	2482	2488	2489	2495	2505
	2506	2512	2513	2519	2529	2530	2536	2537	2543	2553	2554	2560	2561	2567	2577
	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591	2592
	2593	2594	2595	2596	3115	3347	3353	3367	3373	4479	4482	4587	4606	4649	4704
	4759	4974	5047	5048	5049	5367	5405	5451	5582	5583	5611	5657	5658	5838	5877
	5978	6004	6007	6010	6013	6016	6019	6246	6879	7073	7117	7478	7479	7480	7481
	7515	7516	7517	7518	7519	7520	7521	7522	7526	7536	7582	7590	7595	7596	7597

7598	7599	7600	7601	7602	7608	7626	7630	7631	7634	7636	7638	7640	8447	8448
8468	8470	8502	8503	8511	8581	8582	8583	8584	8585	8586	8587	8588	8592	8593
8594	8595	8596	8597	8598	8599	8601	8602	8603	8604	8605	8606	8607	8608	8609
9112	9113	9116	9118	9120	9123	9124	9125	9128	9130	9133	9135	9138	9140	9142
9144	9147	9150	9151	9154	9156	9157	9160	9162	9167	9171	9175	9179	9183	9187
9191	9194	9198	9201	9205	9209	9213	9216	9220	9224	9228	9232	9235	9239	9242
9245	9249	9252	9255											

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*, DERPKC.SEG/SOL/CRF/NL:TOC/DS:ERFZ/PAGNUM=DERPKC.SML, DERPKC.P11
RUN-TIME: 59 94 15 SECONDS
RUN-TIME RATIO: 234/169=1.3
CORE USED: 41K (81 PAGES)

